CONTRÔLE DES PROCÉDURES D'AFFAIRES

par

Saïd Labyad

mémoire présenté au Département de mathématiques
et d'informatique en vue de l'obtention du grade de maître ès sciences (M.Sc.)

FACULTÉ DES SCIENCES
UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, août 1998

Le 07 août 1998 , le jury suivant a accepté ce mémoire dans sa version finale.

date

Président-rapporteur: M. Kacem Zéroual
Département de mathématiques et d'informatique

Membre: M. Richard St-Denis
Département de mathématiques et d'informatique

Membre: M. Marc Frappier
Département de mathématiques et d'informatique

Sommaire

La modélisation des procédures d'affaires est un domaine de recherche récent qui vise à faciliter la compréhension et la communication entre agents humains, à supporter et à améliorer les processus de travail, ainsi que leur gestion, leur automatisation et leur exécution. La plupart des résultats obtenus dans ce domaine concernent des langages de description de procédures d'affaires qui permettent d'instancier un modèle donné dans un environnement cible. En pratique, l'organisation est formée par trois sous-systèmes, à savoir : le système opérant (transforme, produit), le système de pilotage (réfléchit, décide, contrôle) et le système d'information (mémorise, traite, diffuse). Le travail de recherche présenté dans ce mémoire constitue une étude exploratoire qui porte sur l'utilisation de la théorie du contrôle des systèmes à événements discrets dans la modélisation et le contrôle de procédures d'affaires. Nous montrons comment des contrôleurs peuvent être intégrés à une organisation en considérant le système opérant comme un procédé et le système de pilotage comme un contrôleur. Nous montrons aussi à partir de deux exemples comment appliquer des algorithmes de synthèse de contrôleurs propres à la théorie du contrôle des systèmes à événements discrets. Un contrôleur est généré à partir du comportement dynamique des procédures d'affaires et de règles de gestion modélisés à l'aide d'automates ou de réseaux de Petri.

Remerciements

Je tiens à exprimer mes plus vifs remerciements à Monsieur Richard St-Denis, professeur à l'Université de Sherbrooke et mon directeur de recherche, pour m'avoir accueilli dans son équipe et guidé avec compétence ce travail. Ses conseils avisés, ses encouragements constants, ses remarques très pertinentes et son aide matérielle ont grandement facilité la réalisation de ce travail, dont l'aboutissement est pour moi l'occasion de lui exprimer toute ma gratitude.

Je remercie tous les responsables et le personnel de l'ERAC/Tensift. Trop nombreux pour être cités, qu'ils sachent combien je leur suis reconnaissant pour l'ambiance sympathique qu'ils ont su créer et les marques d'amitié qu'ils ont manifestées. Je pense en particulier au directeur de l'ERAC/Tensift.

Je remercie Monsieur Kacem Zéroual, professeur et directeur du Département de mathématiques et d'informatique de l'Université de Sherbrooke, de m'avoir aidé par ses conseils et ses critiques toujours constructives. Qu'il soit assuré de mon amitié.

Table des matières

Liste des tableaux

Liste des figures

Introduction

La modélisation des procédures d'affaires (ou procédures) est un domaine de recherche récent qui vise à faciliter la compréhension et la communication entre agents humains, à supporter et à améliorer les processus de travail, ainsi que leur gestion, leur automatisation et leur exécution. La plupart des résultats obtenus dans ce domaine concernent des langages de description de procédures qui permettent d'instancier un modèle donné dans un environnement cible [16]. Ces approches de modélisation, du fait qu'elles sont orientées vers le génie logiciel, sont souvent mécanistes et ne tiennent pas toujours compte de la coopération entre les agents et des phénomènes socio-cognitifs, même si des tentatives ont été faites dans ce sens [22].

Une procédure dans une organisation fixe ce qu'on fait (objectifs), qui le fait (agents, acteurs ou utilisateurs) et avec quelles ressources (financements, outils), le tout sous des contraintes de coûts, de délais et de qualité. La procédure a pour résultat un produit qui évolue avec les différentes étapes et les acteurs de la procédure. On peut représenter la procédure de deux manières différentes :

- D'une part, en représentant de manière précise la séquence des actions des agents avec leurs liens de précédence, ce qui revient à planifier les actions.

- D'autre part, en représentant les résultats attendus et les objectifs de la procédure de manière à ce qu'elle constitue un domaine consensuel émergeant des actions des agents. Il faut donc pouvoir faire évoluer cette représentation en fonction des actions et de leurs résultats, les actions des agents portant sur les objets de l'environnement.

Problème

L'organisation (entreprise, organisme) est définie dans un premier stade, comme une boîte noire assurant une fonction de transformation d'un flux physique (ou ressources) d'entrée en un flux physique (ou ressources) de sortie. Dans un deuxième stade, cette description devient une boîte blanche, c'est-à-dire que le contenu de la boîte est peu à peu défini. On a donc des constituants de la systémique à savoir :

- Le système opérant, dans un premier temps, est une boîte noire qui prend en compte les flux entrants (flux physiques, flux financiers) et les flux extrants (produits ou services spécifiques à l'activité de l'organisation).

- Le système de pilotage qui regroupe les principales fonctions d'arbitrage et d'allocation de ressources (prévision, planification), de suivi de leur utilisation (contrôle budgétaire, contrôle de gestion) et d'adaptation du fonctionnement de l'organisation à son environnement au travers d'un certain nombre de règles de gestion (contraintes). Trois activités complémentaires y sont donc incluses :

    - la définition des règles de transformation ou de prise en compte des règles édictées par les organisations de tutelle et leur application au fonctionnement de l'organisation et de ses différents services ;

    - l'allocation des ressources nécessaires au fonctionnement des services et la prise de connaissance du compte rendu de leurs activités ;

    - l'ajustement de l'interprétation des règles et de l'utilisation des ressources en fonction des difficultés rencontrées et des écarts constatés entre les objectifs et les résultats obtenus.

- Le système d'information qui est une boîte noire par laquelle transitent les flux d'information, d'une part, entre le système de pilotage et le système opérant (règles de

fonctionnement, ressources allouées, priorités d'exécution) et, d'autre part, entre le système opérant et le système de pilotage (variables de mesure de l'activité et de l'efficacité technique explicitant la variété des résultats obtenus ainsi que la quantité de ressources consommées). Nous distinguons dans le système d'information deux sous-ensembles : l'ensemble structuré des données mémorisées ou mémoire du système d'information et l'ensemble synchronisé des procédures d'affaires.

Le problème abordé dans ce mémoire porte sur le contrôle des procédures d'affaires. L'utilisation d'outils formels de modélisation tels que les automates ou les réseaux de Petri nous permet de décrire formellement les comportements dynamiques des procédures d'affaires et de générer automatiquement des contrôleurs. Notre approche est basée sur la théorie du contrôle des systèmes à événements discrets proposée initialement par Ramadge et Wonham [35].

Dans le cadre de cette théorie, Wonham et Ramadge [48] ont développé un algorithme de synthèse de contrôleurs à partir des langages L (comportement libre du procédé) et K (comportement légal du procédé, $K \subseteq L$). Le langage légal représente les contraintes (règles de gestion de l'organisation) imposées au procédé (le système opérant de l'organisation). Le problème de synthèse de contrôleurs peut être formulé de quatre façons à savoir :

- le cas de base comportant un seul langage légal,
- la synthèse modulaire comportant un langage légal pour chaque contrainte,
- la synthèse avec observation partielle dans laquelle certains événements peuvent être masqués au contrôleur,
- la répartition du contrôle à l'aide de plusieurs contrôleurs agissant d'une façon décentralisée sur le procédé.

Dans ce mémoire, nous restreignons notre étude au cas de base.

Méthodologie

Pour mener à terme notre projet de recherche, nous avons jugé utile de préciser au départ un certain nombre d'objectifs énumérés comme suit :

- faire une recherche bibliographique des travaux théoriques ou pratiques concernant la problématique de synthèse de contrôleurs de procédures d'affaires ;
- étudier les différentes approches de modélisation, jugées utiles, des comportements dynamiques des procédures d'affaires sans se préoccuper dès le départ de leur aspect statique ;
- proposer une architecture d'un modèle dans lequel le procédé correspond au système opérant et le contrôleur agit comme système de pilotage ;
- modéliser le comportement dynamique des procédures d'affaires à l'aide d'automates ou de réseaux de Petri et synthétiser les contrôleurs correspondants.

Pour décrire la dynamique d'une organisation, deux approches générales semblent émerger à savoir : l'approche orientée état et l'approche opérationnelle. Dans l'approche orientée état, une transition est définie comme une paire d'états : état de départ et état d'arrivée. Les actions sont associées aux états et les transitions sont instantanées. L'approche opérationnelle est orientée événement. Elle utilise les concepts d'action élémentaire (comme créer une information, effacer une information) et de transaction, d'une durée non nulle, comme séquence indivisible d'actions élémentaires faisant passer les informations d'un état cohérent à un autre état cohérent. Nous nous rattachons quant à nous à cette deuxième approche en ne mettant pas l'accent sur les informations mais sur la façon dont les différents stimuli apparaissent dans le temps : succession aléatoire ou avec ordonnancement spécifique, mise en évidence de « points de rendez-vous » entre stimuli.

Pour synthétiser des contrôleurs à partir du comportement dynamique des procédures d'affaires et des règles de gestion, nous utilisons deux approches. La première approche

se résume comme suit. À partir d'une procédure d'affaires, en privilégiant le point de vue des acteurs, nous définissons des procédures acteurs afin de mettre en évidence la contribution de chacun d'eux. Une procédure acteur comprend l'ensemble des tâches et des actions effectuées par un même acteur dans le cadre d'une même procédure d'affaires. En utilisant l'approche constructive de Lamport et Lynch [21], nous modélisons le comportement dynamique de chaque procédure acteur ainsi que chaque règle de gestion à l'aide d'un automate. L'automate décrivant le comportement dynamique de la procédure d'affaires est alors obtenu par composition synchrone des automates propres aux procédures acteurs. Le langage légal est aussi représenté par un automate, celui obtenu à partir de l'intersection des automates qui modélisent les règles de gestion. À l'aide de l'outil SUCSEDES [32], nous générons un contrôleur qui, une fois intégré à l'organisation, permet de satisfaire les règles de gestion.

Dans la deuxième approche, la modélisation des comportements dynamiques des procédures d'affaires est réalisée à l'aide des réseaux de Petri. Puisque cette approche présente des difficultés, nous avons jugé utile de passer par un formalisme informel et intuitif pour l'utilisateur, le modèle organisationnel des traitements (MOT) de la méthode Merise. le MOT constitue une vision plus globale d'un système d'information par procédures d'affaires. Le MOT intègre les notions de temps et de durée, de ressource, de lien et de nature du traitement. Le regroupement des différentes procédures d'affaires donne le MOT qui est ensuite traduit en un réseau de Petri. Pour la synthèse du contrôleur, nous utilisons les réseaux de Petri avec des places d'entrée externes.

Le point de départ de cette deuxième approche est constitué des éléments obtenus lors du recueil de l'existant et particulièrement le schéma des flux d'informations ainsi que le découpage du domaine en processus. Pour chaque processus identifié, nous établissons dans un premier temps un diagramme de circulation des documents en nous servant des informations contenues dans le schéma des flux. À partir des diagrammes de circulation des documents, nous établissons les procédures d'affaires par élimination des ressources

permanentes et par introduction des conditions de déclenchement des traitements. Dans ces procédures d'affaires, les acteurs et le temps apparaissent toujours. L'ensemble des procédures d'affaires obtenu constitue le modèle organisationnel des traitements de l'organisation.

Les réseaux de Petri nous intéressent pour au moins trois raisons : l'existence d'une représentation graphique simple en termes de transitions et de places à travers lesquelles circulent des jetons, leur adaptation possible à la simulation discrète et leurs bases théoriques mathématiques.

Résultats

Après un examen de la littérature relative au contrôle des procédures d'affaires et selon notre connaissance, nous pouvons affirmer qu'il n'existe pas de travaux portant sur le problème de la synthèse de contrôleurs de procédures d'affaires, d'où l'originalité de notre travail de recherche. La plupart des résultats obtenus dans le domaine concernent des langages de description des procédures d'affaires.

Dans le cadre de ce projet de recherche, nous montrons comment les contrôleurs peuvent être intégrés à une organisation en substituant le système de pilotage au contrôleur et le système opérant au procédé. Nous montrons aussi à l'aide de deux exemples comment synthétiser des contrôleurs à l'aide d'algorithmes propres à la théorie du contrôle des systèmes à événements discrets.

Le premier exemple porte sur une bibliothèque et utilise les automates comme outil formel de modélisation des comportements dynamiques d'une procédure d'affaires et des règles de gestion. Le deuxième exemple porte sur le traitement de devis et utilise les réseaux de Petri comme outil formel de modélisation et le MOT de la méthode Merise.

Organisation du mémoire

Dans le chapitre 1, nous donnons une idée précise de la modélisation des comportements dynamiques des procédures d'affaires selon trois méthodes (Merise, OSSAD et OMT) et de quelques outils d'analyse des procédures d'affaires. Le chapitre 2 porte sur le contrôle des procédures d'affaires. Il contient la formulation du problème de contrôle des procédures d'affaires et la description des méthodes de synthèse de contrôleurs. Les chapitres 3 et 4 présentent deux exemples complets qui illustrent l'utilisation de notre approche, tout d'abord avec les automates, puis avec les réseaux de Petri. Enfin, nous concluons le mémoire avec une présentation des contributions, des critiques et des extentions possibles de notre travail.

Chapitre 1

Procédures d'affaires

Une procédure d'affaires est un ensemble coordonné (en série ou en parallèle) d'activités manuelles ou informatisées qui sont connectées dans le but d'atteindre un objectif commun. L'instance d'une activité ou tâche est associée à un acteur, généralement possédant le rôle spécifié dans l'activité. Le rôle est la position que peut prendre un acteur et à laquelle peuvent correspondre les activités que cette position permet d'utiliser. Les procédures et les tâches permettent d'introduire de façon naturelle une modularité dans l'organisation [24].

Les nouveaux modes de communication constituent un enjeu stratégique pour les organisations. L'organisation du travail s'appuie désormais sur les technologies de l'information qui deviennent déterminantes dans la profitabilité des organisations. Des travaux théoriques, tels que ceux d'Ellis associés aux approches industrielles, ont donné naissance aux premiers outils workflow (ou workflows) [15].

Il existe plusieurs définitions relatives aux workflows. Par exemple, Fischer et While définissent le workflow comme étant une séquence d'actions ou d'étapes utilisées dans les procédures d'affaires [47]. Hollingsworth considère les workflows comme des modèles informatisés des procédures d'affaires [19]. Le workflow management est un important outil pour structurer et optimiser les traitements d'affaires et pour supporter l'implémentation

pratique de la réingénierie des procédures d'affaires [17].

En l'absence de consensus sur la définition du workflow, il y a une certaine confusion et un mélange des concepts avec ceux du *groupware* (collectitiel) et de la *GED* (Gestion Électronique de Documents).

Dans le cadre de ce mémoire, nous utilisons comme définition du workflow celle d'outil décisionnel coopératif dont les paramètres sont un nombre limité de personnes devant accomplir en un temps limité des tâches articulées autour d'une procédure d'affaires définie et ayant un objectif global [42]. Aujourd'hui encore, toutes les techniques de workflow sont sous-tendues aux concepts industriels, notamment l'optimisation des procédures d'affaires.

À titre d'exemple, nous décrivons une spécification générale d'un système appelé W3flow [3] [20] [37]. Il est considéré comme une extension du workflow supportant une coopération flexible et une coordination entre les organisations. L'objectif n'est pas de donner une description complète mais seulement une perception du système W3flow.

La connaissance générique est le noyau du système W3flow. Elle est organisée sous forme de modèles qui sont des abstractions des aspects bien définis du domaine de coopération [20].

Modèle inter-organisationnel Le modèle inter-organisationnel définit le qui. Pour une entité organisationnelle, ce modèle représente des informations concernant la structure inter-organisationnelle (groupe, organisation, acteur) [1] [29]. Il représente aussi les informations concernant le rôle tenu par chaque acteur.

Modèle d'information Le modèle d'information décrit les différents types d'information qui sont des entrées ou sorties des tâches et leurs relations avec les entités (objet physiques ou abstraits ayant des caractéristiques comparables) [2] [7] [34].

Modèle conversationnel Le modèle conversationnel définit le quoi et le comment. Il représente l'échange de messages entre les acteurs pour l'accomplissement des tâches [33].

Modèle du temps Le modèle du temps est la base pour coordonner les actions issues du processus inter-organisationnel. Il est essentiel pour représenter le délai, l'achèvement de la tâche et le début de la tâche [46].

Modèle d'exécution Le modèle d'exécution permet l'exécution des workflows inter-organisationnel. Il programme la tâche à exécuter par l'acteur et gère les questions et les conversations qui y sont apparentées.

Les procédures d'affaires sont représentées par le modèle inter-organisationnel du W3flow. Les informations portées par les événements déclencheurs des procédures d'affaires sont décrites dans le modèle d'information. Les échanges d'informations entre les postes de travail (acteurs) des procédures d'affaires sont décrits par le modèle conversationnel. Le temps relatif au début et à la fin des procédures d'affaires est représenté par le modèle du temps. Enfin, elles sont exécutées par le modèle d'exécution.

Le W3flow, à l'aide de ses modèles, permet la représentation des procédures d'affaires en termes d'acteurs, de groupes d'acteurs et de rôles tenus par chaque acteur (modèle inter-organisationnel) jusqu'à leurs exécutions par le modèle d'exécution.

Il existe d'autres approches qui modélisent les procédures d'affaires. L'approche « OSSAD » (*Office Support Systems Analysis and Design*), grâce à ses modèles (abstrait, descriptif et perscriptif), modélise les procédures d'affaires en termes de fonctions, de rôles et de documents. La méthode « OMT » (*Object Modeling Technique*), grâce à son modèle dynamique, modélise les procédures d'affaires en termes d'événements, d'états et d'opérations. Enfin, la méthode Merise, grâce à ses modèles conceptuels et organisationnels de traitements, modélise les procédures d'affaires en termes de procédures fonctionnelles ou d'opérations organisationnelles. En plus, elle permet une vérification formelle des comportements des procédures d'affaires à l'aide d'analyses des réseaux de Petri. Une étude comparative entre la terminologie de Merise et celle de Workflow est donnée en annexe A. Les concepts présentés dans ce mémoire sont principalement tirés des références [4] [12] [23] [26] [31] [44].

1.1 Les procédures d'affaires dans Merise

La modélisation des comportements dynamiques des procédures d'affaires dans la méthode Merise [44] s'exprime dans un formalisme spécifique, élaboré pour permettre de représenter le fonctionnement d'activités aux différents niveaux de préoccupations (conceptuel, organisationnel, logique, physique). Il repose sur des bases théoriques solides permettant une vérification formelle des modèles dans la mesure où il s'inspire du formalisme des réseaux de Petri.

Dans le cadre de ce mémoire, seuls les modèles conceptuels et organisationnels sont utilisés. Pour décrire le niveau conceptuel, le formalisme des traitements comporte les concepts d'événement, de résultat, d'opération et de synchronisation.

1.1.1 Modèle conceptuel des traitements

L'activité d'une organisation se décompose en procédures d'affaires, généralement indépendantes des choix d'organisation, qui s'articulent en opérations déclenchées par des événements soumis à une synchronisation pour produire des résultats.

Le modèle conceptuel des traitements (MCT) doit symboliser l'activité de l'organisation, avec les procédures d'affaires et règles de gestion associées, en faisant abstraction des ressources humaines et matérielles ainsi que leur mode d'exploitation (centralisée, décentralisée, distribuée, manuelle, automatisée).

CONCEPTS

Opération Une opération est un ensemble structuré de règles de gestion se traduisant en traitements arithmétiques ou logiques qui se déroulent sans attente externe du domaine d'étude en réaction à un stimulus. Elle est déclenchée soit par un événement unique soit par une synchronisation de plusieurs événements. La description de l'opération correspond essentiellement à la description des vues du gestionnaire. Que vérifie-t-on? Que calcule-t-on? Que met-on à jour?

Exemples : vérification d'une demande de prêt et calcul des intérêts cumulés.

Pour pouvoir traiter l'occurrence d'un événement, l'opération doit disposer de toutes les ressources nécessaires (données et moyens). Dans un système d'information, plusieurs opérations peuvent être activées au même instant.

Les règles de gestion formalisent les comportements répétitifs de l'organisation. Elles sont l'expression littéraire des traitements logiques ou arithmétiques qui composent une opération.

On utilisera un langage simple, précis et non ambigu pour les règles de gestion en mettant en évidence les actions portant sur le système d'information (recherche, création et modification d'informations).

Les règles de gestion d'un système d'information ont deux types d'origine :

Origine interne à l'organisation Les règles prises par les gestionnaires de l'organisation (en termes d'analyse de système). Elles font partie du système de pilotage et elles sont susceptibles d'évoluer dans le cadre du nouveau système à réaliser.

Origine externe à l'organisation Les règles prennent naissance dans un texte de loi ou d'un accord entre plusieurs organisations. Elles sont exprimées de façon précise et elles doivent souvent être considérées comme imposées dans le cadre du nouveau système.

Événement L'événement est un fait actif, prévu par l'organisation pour le système d'information provoquant une réaction matérialisée par une succession d'application de règles de gestion et concourant à produire un ou plusieurs résultats.

Un événement représente un fait pour le système d'information ; il est généralement porteur d'informations et peut être considéré comme un message ; il est doté

d'un même ensemble de propriétés ; il n'est présent qu'une fois dans le système d'information ; chaque occurrence d'un événement est identifiable.

On distingue trois types d'événement :

Événement externe On appelle événement externe, tout événement émis par l'environnement du système. Un événement externe est un stimulus provoquant une réaction du système. L'événement externe est provoqué soit par un acteur externe à l'organisation (banque, client), soit par un acteur interne à l'organisation mais hors du champ d'étude.

Exemples : réception d'une commande, réception d'un avis de crédit, accord du comptable.

Événement interne On appelle événement interne tout événement apparaissant au sein du système. Il est issu d'une opération du système.

Exemples : commande enregistrée, facture en attente de paiement.

Événement artificiel Les événements ne provenant pas d'acteurs externes ou internes sont dits artificiels. Ils dépendent des règles de gestion, des contraintes légales ou des contraintes naturelles. À un événement artificiel sont généralement rattachées les notions de temps ou de niveau.

Exemples : fin de mois, à la date d'échéance, à l'atteinte du niveau.

Synchronisation D'une façon générale, une opération est activée par la réalisation d'une condition de synchronisation entre plusieurs événements. Cette condition s'exprime par une fonction booléenne des événements concourant à l'opération.

Résultat Le résultat est la réponse codifiée d'une opération déclenchée par un événement ou par une synchronisation d'événements. Le résultat possède les mêmes caractéristiques que l'événement. Il peut constituer à son tour un événement pour une autre opération du système.

On distingue le résultat interne (à synchroniser dans une autre opération) du résultat final qui correspond à un état final du phénomène étudié pour le champ d'étude.

PRINCIPES DE CONSTRUCTION DU « MCT »

Non-interruptibilité Une opération conceptuelle est dite non interruptible quand elle dispose, après la synchronisation des événements qui la déclenchent, de toutes les ressources nécessaires pour son exécution complète.

La non-interruptibilité interdit l'arrêt du traitement d'une occurrence d'événement soit par l'arrivée d'une autre occurrence d'événement du même type ou d'autres occurrences d'événements inclus ou non dans la synchronisation rattachée à l'opération.

Non-redondance Le principe de non-redondance implique que les règles de gestion du champ d'étude sont spécifiques à chaque opération figurant dans le modèle. La redondance des traitements consiste à exécuter à différents endroits du modèle une ou plusieurs règles de gestion décrivant une opération. Elle est évitée par une modification des synchronisations des opérations.

DÉMARCHE POUR LA CONSTRUCTION DU « MCT »

Voici les principales étapes à réaliser pour la construction du MCT :

- rechercher les acteurs impliqués (externes à l'organisation, internes mais hors du champ d'étude et internes dans le champ d'étude) ;
- rechercher les *FLUX* entre acteurs ;
- identifier les *ÉVÉNEMENTS* ;
- identifier les *RÉSULTATS* ;
- identifier et décrire les *OPÉRATIONS* et les *RÈGLES DE GESTION* ;
- déterminer les *SYNCHRONISATIONS* pour chaque *OPÉRATION* ;
- rechercher l'enchaînement des *OPÉRATIONS* et préciser les conditions d'émission des *RÉSULTATS* ;
- appliquer les règles de *NON-REDONDANCE* et de *NON-INTERRUPTIBILITÉ* pour la vérification du modèle. Si nécessaire, revoir les synchronisations.

La représentation graphique du modèle conceptuel des traitements est présentée à la figure 1.

1.1.2 Modèle organisationnel des traitements

Le modèle conceptuel des traitements a permis de décrire les fonctions majeures du domaine sans référence aux ressources nécessaires pour en assurer le fonctionnement. Il se concentre sur le quoi et le pourquoi. Le modèle organisationnel des traitements se concentre sur le comment.

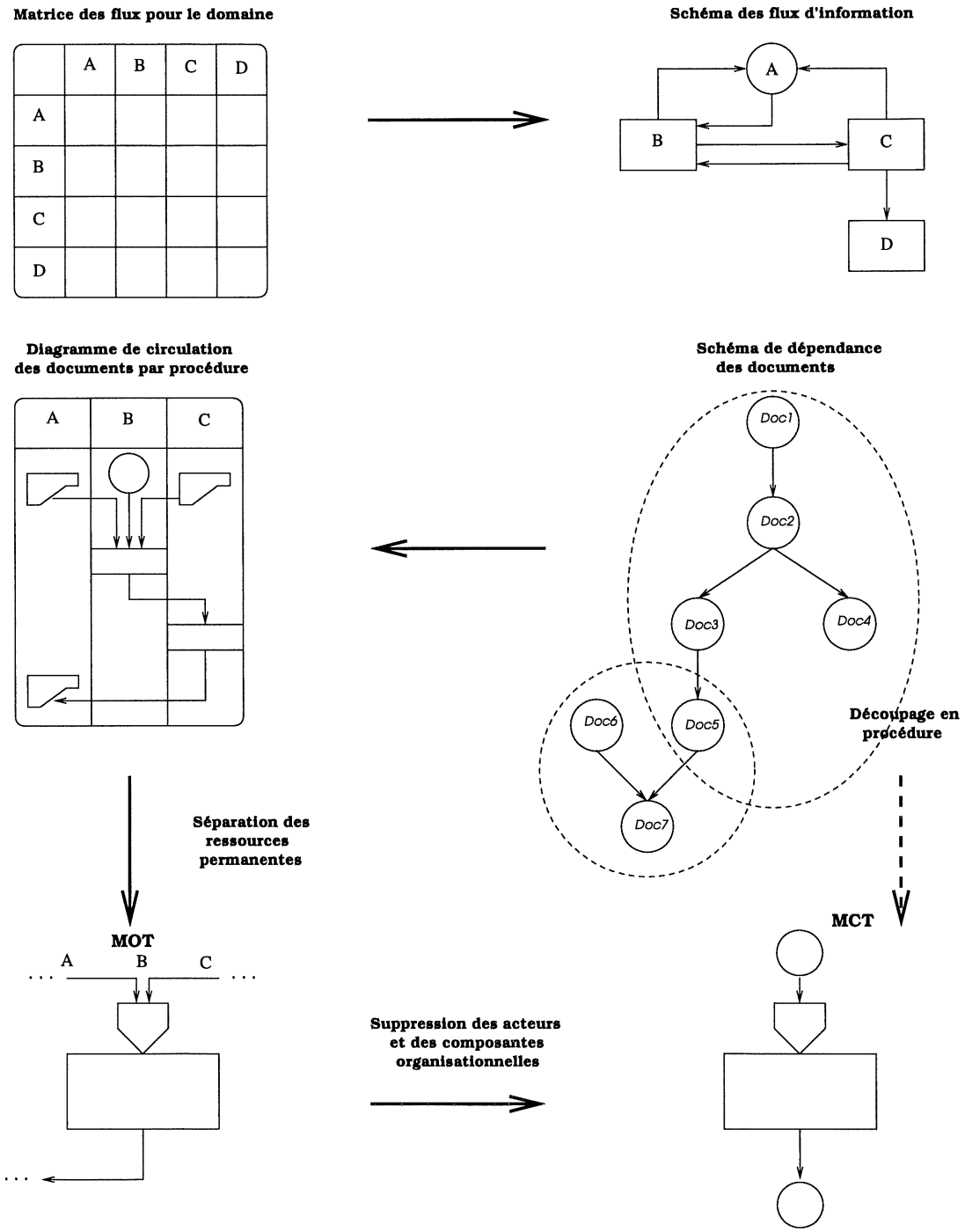

FIG. 1 – *Élaboration des modèles de traitements*

Le modèle organisationnel des traitements (MOT) symbolise l'activité de l'organisation. Il exprime la façon dont les règles de gestion sont appliquées dans l'organisation et indique l'utilisation des ressources mises en œuvre lors de cette application.

Le modèle décrit l'organisation du système d'information : circuits, postes de travail, degré d'automatisation des traitements, leur enchaînement dans le temps et dans l'espace, répartition des tâches entre l'homme et la machine, sécurité.

Il ne dépend pas du choix définitif des moyens matériels et logiciels spécifiques qui seront utilisés au niveau opérationnel. Les préoccupations essentielles sont : *QUI* fait *QUOI*? *QUAND*? et *OÙ*?

La représentation graphique du modèle organisationnel des traitements est analogue à celle du modèle conceptuel des traitements (voir figure 1).

CONCEPTS

Procédure fonctionnelle ou opération organisationnelle Une procédure fonctionnelle est un ensemble logique structuré d'actions élémentaires dont l'activation est provoquée par l'apparition d'un ou plusieurs événements. Elle est réalisée par un même acteur et son déroulement ne nécessite pas l'intervention d'autres événements ou acteurs.

- Elle est caractérisée par un ensemble logique déterminé et structuré d'actions élémentaires, un type de poste de travail déterminé et une période déterminée pour son exécution.
- Elle est déclenchée par l'arrivée d'un événement ou par la synchronisation de plusieurs événements.
- Elle peut être manuelle ou automatisée.

Généralement, une opération conceptuelle donnera lieu à une ou plusieurs procédures fonctionnelles.

Action élémentaire Au sein d'une opération, l'action élémentaire est le plus petit traitement significatif par rapport au niveau d'analyse retenu ; elle s'effectue sans interruption.

Exemples : saisie d'une donnée, vérification de la valeur prise par une donnée, confirmation par un opérateur d'une valeur affichée à l'écran.

Les actions élémentaires seront définies à partir des règles de gestion identifiées au niveau conceptuel. L'exécution d'une règle de gestion générera une ou plusieurs actions élémentaires.

Acteur Un acteur est une personne qui prépare, transmet, transforme, exploite ou reçoit les données d'un système d'information. Il peut être interne au champ d'étude (opérateurs), interne à l'organisation mais hors du champ d'étude (comptable recevant les doubles des commandes), externe à l'organisation (client).

Enchaînement de procédures On regroupera souvent les procédures dans des enchaînements correspondant à des ensembles structurés d'opérations concourant à l'élaboration d'un ou plusieurs résultats en réponse à la sollicitation d'une famille d'événements caractéristiques d'une activité importante de l'organisation étudiée, du fait des lois et des fins propres de cette organisation.

Poste de travail Le poste de travail est un centre d'activité élémentaire et opérationnel de l'organisation comprenant tout ce qui est nécessaire (hommes, machines, espace, outillage) à l'exécution des traitements définis, automatisés ou non.

Événements ou résultats Les événements ou résultats seront décrits en précisant la nature de l'événement ou du résultat ; les besoins de sécurité ; l'origine ou la destination fonctionnelle et géographique ; la fréquence d'apparition ou d'émission sur une période donnée ; le temps de réaction entre l'apparition de l'événement et le moment où il est traité et les problèmes de stockage liés à ce décalage.

PRINCIPES DE CONSTRUCTION DU « MOT »

Non-redondance des opérations Il y aura redondance si deux opérations comportant les mêmes actions élémentaires s'effectuent au même poste de travail et au même moment pour traiter le même ou les différents types d'événements. Il convient alors de modifier la synchronisation des événements qui déclenchent l'opération de manière à ce qu'elle reste unique.

Non-interruptibilité La procédure fonctionnelle est non interruptible comme l'opération conceptuelle. Cependant, le traitement d'une occurrence pourra être interrompu par l'exécution, au même moment, d'une procédure plus prioritaire.

Exemple d'une procédure interruptible : la dactylographie d'une lettre peut être interrompue momentanément par la nécessité de répondre au téléphone.

DÉMARCHE POUR LA CONSTRUCTION DU « MOT »

La démarche pour la construction du MOT comporte deux parties principales :

- Établir le modèle organisationnel des traitements de l'existant (effectuer des entrevues ; établir la matrice des flux à partir des entrevues et des informations recueillies concernant l'enchaînement des opérations ; décrire les enchaînements de procédures et les procédures elles-mêmes, la description des opérations se fera en utilisant le formalisme et les fréquences et les volumes de traitement pour chaque opération ; analyser les postes de travail).

  Le modèle organisationnel de traitements comprendra, à ce stade, le modèle d'enchaînement et une description succincte des opérations.

  Cependant, les opérations comportant un « passif » important ou perçues comme stratégiques gagneront à être décrites plus en détail.

- Élaborer le ou les modèles organisationnels des traitements des variantes futures ou du système futur à partir du modèle conceptuel des traitements. Les règles de gestion ont été définies pour chaque opération recensée lors de l'élaboration du modèle conceptuel de traitement.

  Le passage au niveau organisationnel implique deux décisions essentielles : affectation d'un poste de travail et choix du degré d'automatisation. Ce passage se fera en lisant les règles de gestion et en étudiant les contraintes organisationnelles pour chaque règle.

  Les principales contraintes sont fonction :

  - de la possibilité de formuler le problème en termes arithmétiques ou logiques ;
  - de la possibilité de traduire sous forme algorithmique l'application de la règle ;
  - du degré d'urgence d'exécution ;
  - du volume à traiter sur une période précise ;
  - des contraintes ergonomiques ;
  - des contraintes psychologiques et sociales ;
  - d'une recherche de fiabilité offerte par l'automatisation ;
  - du coût et des contraintes de budget ;
  - de toutes autres contraintes découlant de la politique générale de l'organisation.

  Il sera difficile d'être exhaustif, chaque projet possédant ses spécifications, mais il est souhaitable d'avoir une vue synthétique du problème traité.

  Les opérations sont éclatées en règles de gestion et sont étudiées individuellement. De plus, les règles de gestion qui peuvent s'exécuter (au même poste de travail ; au

même moment et suivant le même degré d'automatisation) pourront être regroupées au sein des mêmes procédures fonctionnelles.

On aboutit ainsi à la démarche suivante pour chacune des variantes étudiées :

- identifier les modifications de structures correspondantes : fonctions, sous-fonctions, activités ;
- recenser les règles de gestion issues du modèle conceptuel des traitements ;
- étudier l'opportunité d'automatiser chaque règle de gestion en tenant compte des contraintes d'organisation et des critères de succès ;
- déterminer les postes de travail qui exécuteront les règles de gestion ;
- regrouper les règles de gestion en procédures en respectant les principes de construction ;
- définir les enchaînements entre procédures en faisant apparaître les documents échangés entre les postes de travail ;
- définir succinctement les actions élémentaires de chaque opération ;
- faire la synthèse des ressources nécessaires (humaines et matérielles) ;
- vérifier que le modèle obtenu satisfait aux objectifs d'organisation et aux critères de succès, notamment en ce qui concerne les délais, les temps de réponse, la qualité de service.

Il est à noter que les événements et résultats du modèle conceptuel de traitement devront être complétés par des événements ou résultats issus des contraintes organisationnelles.

1.2 Les procédures d'affaires dans « OSSAD »

OSSAD (*Office Support Systems Analysis and Design*) résulte d'un projet du programme ESPRIT (*European Strategic Program for Research in Information Technology*) conduit de 1985 à 1990 par une équipe multinationale de consultants, d'universitaires et d'usagers des technologies de l'information [11] [14]. OSSAD est à la fois une démarche et un ensemble d'outils de modélisation pour soutenir cette démarche.

Le comportement dynamique des procédures d'affaires dans OSSAD est représenté à l'aide des modèles abstrait, descriptif et perscriptif respectivement en termes de fonctions, rôles et documents.

1.2.1 Outils de modélisation

OSSAD préconise une double modélisation : **abstraite** (qui s'attache au « quoi » et au « pourquoi » de l'organisation) et **descriptive** (qui s'attache au « comment » sont réalisées les missions de l'organisation et au « qui » y collabore).

Les modèles abstraits permettent de représenter les objectifs de l'organisation étudiée à partir des deux concepts de **fonction** (symbolisé par un rectangle) et de **paquet** d'information (symbolisé par une ellipse). Par des *zooms* successifs sur les fonctions on aboutit à des **activités** qui sont par définition des fonctions qui ne sont plus décomposables. On cadre ainsi le problème à résoudre de façon systémique.

Les modèles descriptifs permettent de représenter les moyens matériels mis en œuvre pour atteindre les objectifs modélisés au niveau abstrait et les responsabilités des personnes intervenant dans l'organisation (moyens humains). Ils utilisent les concepts de **rôle** et d'**unité** (tous deux symbolisés par un cercle), d'**opération** (un carré), d'**outil** (un triangle) et de **ressource** en informatique (un « casier »). Ces concepts descriptifs permettent de modéliser les structures, les technologies et les ressources humaines à l'œuvre dans toute organisation.

L'ordre dans lequel peuvent être faits les deux types de modèles n'est pas prescrit. Il dépend des interlocuteurs et du problème. Une matrice activités-rôles fait le lien entre les modèles abstraits et descriptifs. Elle permet d'identifier les **tâches** (intersection entre un rôle et une activité) de l'organisation étudiée.

1.2.2 Modélisation du réel

L'approche OSSAD distingue deux niveaux d'abstraction ou de modélisation :

- l'**abstrait** pour exprimer les **objectifs** d'une organisation (le « quoi » et le « pourquoi »),
- le **descriptif** pour exprimer ses **moyens** (humains et technologiques).

Pour chacun de ces niveaux (voir figure 3), OSSAD propose un formalisme (voir figure 2), c'est-à-dire un ensemble de concepts et de règles permettant de structurer les représentations pour mieux communiquer. À ces niveaux d'abstraction correspondent deux grands concepts sous-jacents :

- la **fonction** qui est le concept de base du modèle abstrait,
- le **rôle** qui est le concept de base du modèle descriptif.

Un troisième niveau, dit **perscriptif**, permet de faire le pont, si cela est nécessaire, avec des méthodes et outils de développement d'application informatique.

Le formalisme *ossadien* est **graphique**. Les modèles sont représentés sous forme de **graphes**. Par la suite, les mots « graphe » et « modèle » seront employés indifféremment. En toute rigueur, un modèle peut être constitué d'un ou plusieurs graphes de même type. Un graphe se compose de nœuds et de liens entre ces nœuds (voir figure 4). Le dessin de ces graphes est régi par des règles de grammaire *ossadiennes*, concernant principalement :

- la forme graphique des nœuds (un nœud est la représentation icônique d'un objet

de la modélisation, chaque concept *ossadien* possédant une forme graphique bien déterminée) ;

- les règles lexicographiques pour la dénomination des objets ;
- la forme graphique des liens (type de flèche, couleur, type de trait) ;
- les types de liens autorisés entre les nœuds ;
- les enchaînements possibles entre les différents graphes.

1.2.3 Exemple

Une organisation est composée de trois fonctions d'affaires qui sont : GESTION, VENTE et PRODUCTION. Dans notre exemple, nous détaillons seulement la fonction d'affaires VENTE. La fonction VENTE englobe trois activités qui sont la gestion des contrats, la facturation et la gestion des commandes. La première activité est assumée par le responsable avant-vente et les deux autres activités sont assumées par le responsable de la clientèle. Le client formule sa commande par télécopieur ou par téléphone au responsable de commande qui la vérifie et l'envoie à l'équipe de facturation pour établir la facture et la transmettre au client.

- **modèle abstrait :** comment représenter les objectifs de l'organisation (voir figures 5 et 6) .
- **matrice activités-rôles :** comment représenter la répartition des moyens humains pour atteindre les objectifs (voir figure 7).
- **graphe de rôles et graphe de procédures :** comment représenter la circulation des informations pour atteindre les objectifs (voir figures 8 et 9).

- **graphe d'opérations :** comment décrire avec précision l'ensemble des moyens (humains et matériels) et le déroulement des procédures pour atteindre les objectifs (voir figure 10).

1.3 Les procédures d'affaires dans « OMT »

Le comportement dynamique des procédures d'affaires dans « OMT » représente l'évolution du système dans le temps, en termes d'**événements** et d'**états**. L'événement correspond à un stimuli externe du système et l'état correspond à la valeur des objets (valeur des composants). Les événements et les états permettent de spécifier le comportement d'un système sous forme d'un diagramme d'états représentant les transitions possibles en fonction des événements (modèle temporel).

Événement/État L'état d'un objet représente les valeurs de ses attributs et de ses liens dans l'intervalle séparant l'arrivée de deux événements successifs sur un objet. L'évolution des objets est spécifiée par des diagrammes d'états qui représentent la succession des états par lesquels passe un objet suite à l'occurence d'événements (voir figure 11). Les transitions d'états peuvent être conditionnelles, auquel cas elles sont exprimées sous la forme de conditions booléennes.

Opérations Les spécifications du comportement d'un objet permet de décrire les opérations à exécuter en présence d'un événement. L'exécution d'actions (opérations élémentaires) constitue la réponse de l'objet à l'événement. Ces actions peuvent correspondre soit à des opérations de changement d'état (modification), soit à des envois de message à d'autres objets. « OMT » introduit aussi la notion d'**activité**. Celle-ci est associée à un état et représente une séquence logique d'actions indissociables avec un début et une fin.

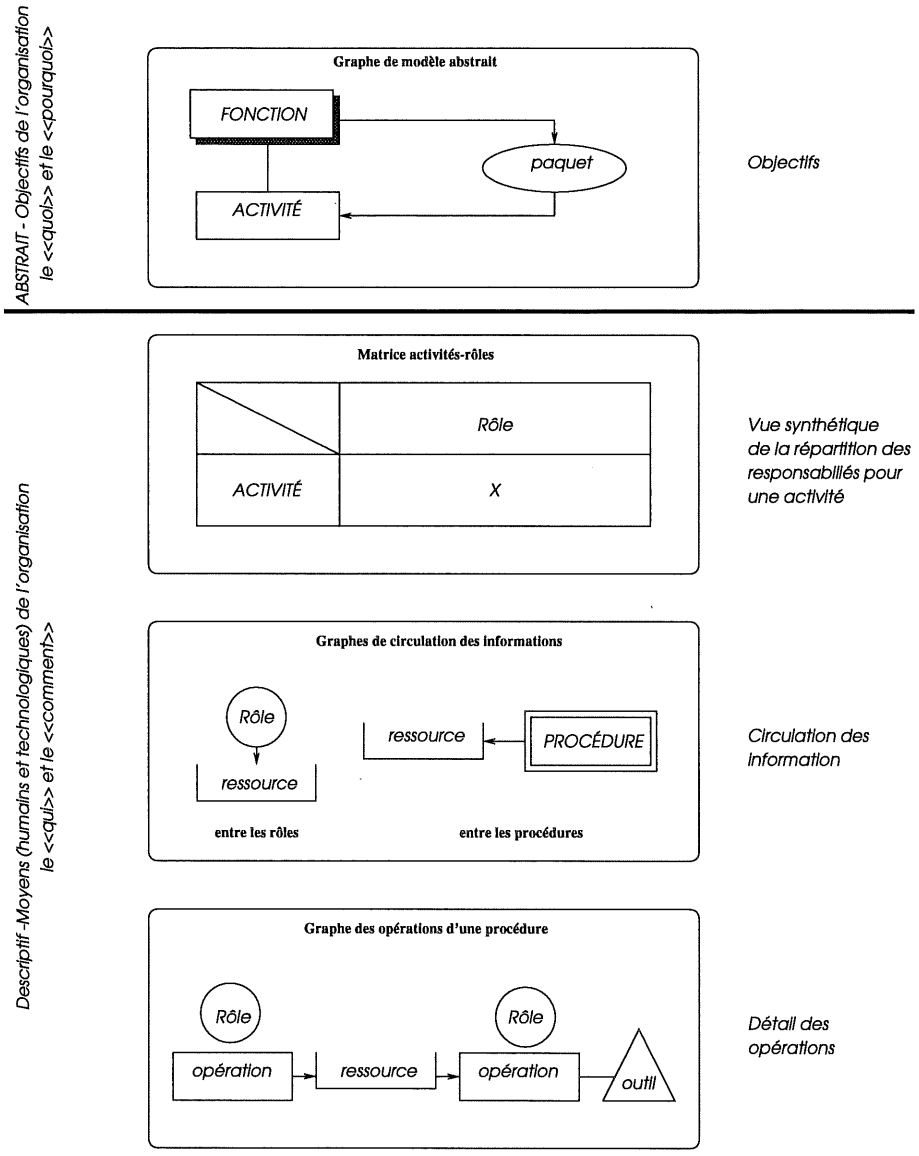

FIG. 2 – *Formalisme graphique*

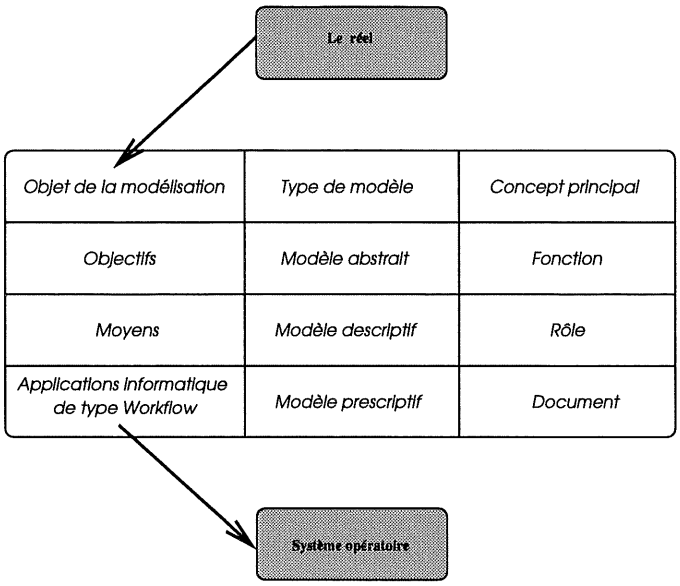

FIG. 3 – *Niveaux de modélisation ossadiens*

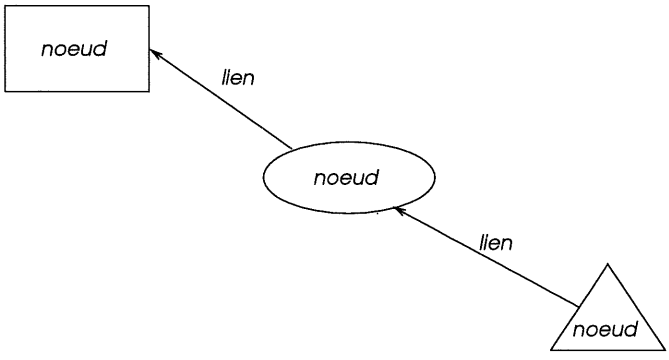

FIG. 4 – *N œuds et liens*

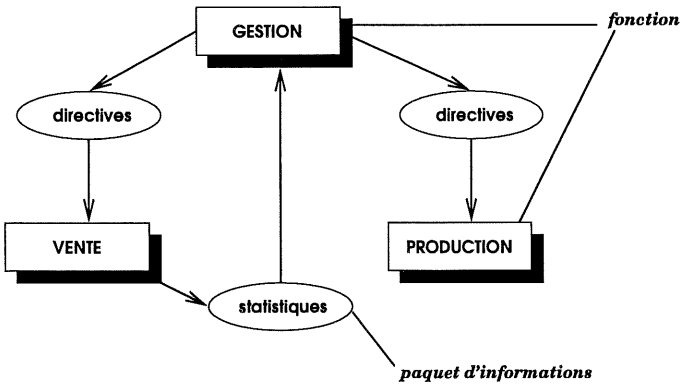

FIG. 5 – *Modèle abstrait*

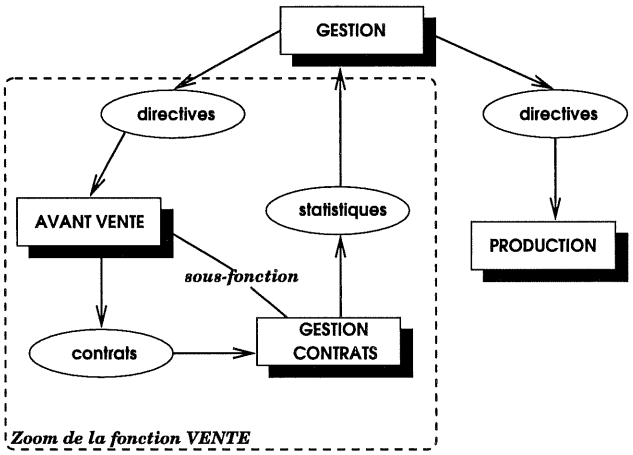

FIG. 6 – *Modèle abstrait avec zoom*

Rôles / Activités	Responsable avant-vente	Responsable clientèle
GESTION CONTRATS	X	
FACTURATION		X
GESTION COMMANDES		X

FIG. 7 – *Matrice activités-rôles*

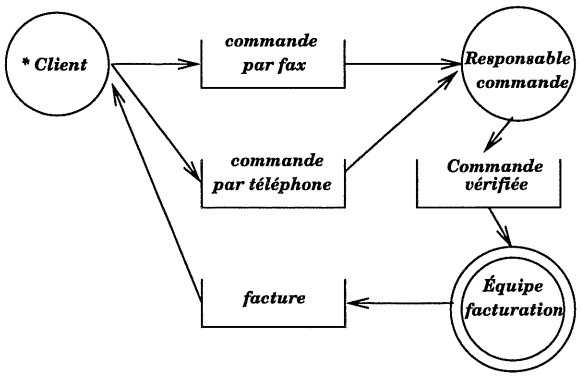

FIG. 8 – *Graphe de rôles*

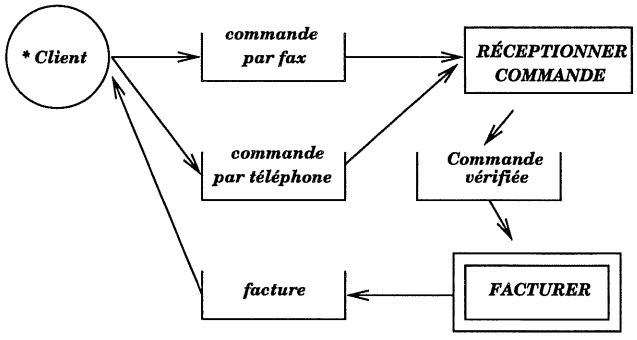

FIG. 9 – *Graphe de procédures*

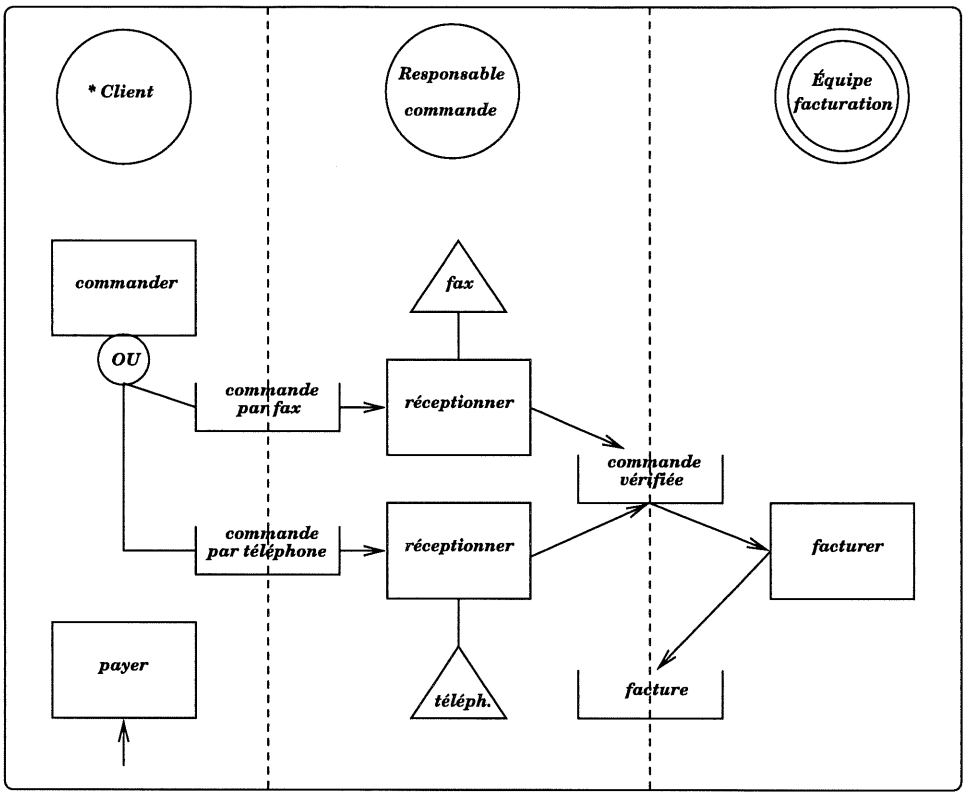

FIG. 10 – *Graphe d'opérations*

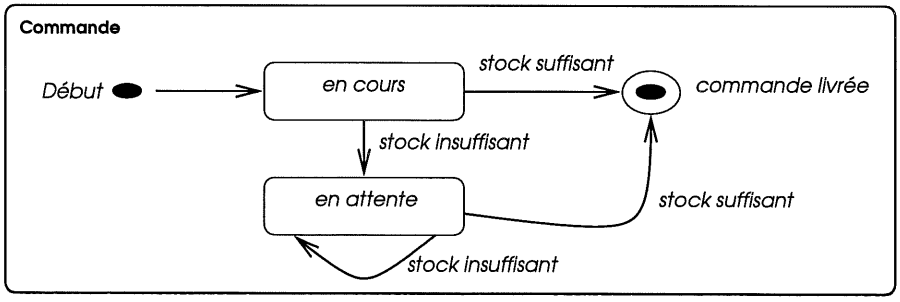

FIG. 11 – *Diagrammes d'états de la classe commande*

1.4 Outils d'analyse des procédures d'affaires

La modélisation du comportement dynamique des procédures d'affaires sous la forme de diagrammes permet leur analyse. Plus riche et formelle est la sémantique rattachée aux diagrammes, plus il est aisé d'automatiser l'analyse des procédures d'affaires afin de dégager les propriétés qu'elles satisfont.

Par exemple, les réseaux de Petri (RdP) permettent de tenir compte du comportement dynamique du système et sont utilisés dans des modèles tels que Merise. Les RdP fournissent un formalisme graphique (plus précis et formel que les diagrammes de flux de données par exemple) donnant un outil de communication fin entre le concepteurs et utilisateurs.

Chaque comportement d'une procédure d'affaire peut être modélisé sous forme d'un réseau de Petri. Un réseau de Petri sera utilisé pour représenter la synchronisation et le découpage des tâches engendrées par un traitement. Un réseau de Petri permet de représenter le fait qu'il existe :

- des tâches qui doivent être effectuées avant d'autres ;
- des tâches qui ne peuvent être effectuées que si certaines autres ont été accomplies ;
- des tâches pouvant être amorcées pendant que d'autres sont en cours.

Cependant, le passage du formalisme de traitement à un réseau de Petri engendre une difficulté qui provient de ce qu'une transition correspond soit à un type d'opération soit à un type de synchronisation. Une transition exprimant un type de synchronisation n'a pas d'action sur l'environnement et a un prédicat associé toujours vrai. Une transition exprimant un type d'opération a une action éventuelle sur l'environnement (système d'information) et le prédicat est équivalent au prédicat de production des résultats.

Notons que le formalisme de traitement a été initialement conçu en s'inspirant du formalisme développé dans les RdP ; l'utilisation importante qui en a été faite récemment

amène un certain nombre de questions sur les analogies entre ces deux formalismes (voir TAB. 1).

La référence théorique aux RdP permet une analyse du comportement des procédures d'affaires en termes de simulation et d'étude de propriétés. L'analyse du graphe d'accessibilité d'un réseaux de Petri permet d'établir si une procédure d'affaires possède ou non un ensemble de propriétés (sûreté, borné, conservateur, vivacité, accessibilité, équivalence et recouvrabilité) [8] [18] [43].

Une analyse par simulation (par exemple à l'aide des outils Bodart et IDA) [7] permet d'approcher le dimensionnement des ressources, la recherche des files d'attente, l'estimation de la durée des procédures. Cependant, pour qu'une telle simulation soit efficace, il est nécessaire de préciser un grand nombre de paramètres dont l'appréciation reste parfois trop subjective au cours d'une étude.

Les approches ci-dessous sont basées sur les notions de rôle et d'activité. Une activité étant un ensemble de tâches qui concourent à un objectif commun et dans laquelle des agents accomplissent des rôles différents.

Formalisme de traitement	**Réseau de Petri**
Type de message	Place
Type d'opération	Transition
Type de synchronisation	Transition
Événement ou résultat	Jeton

TAB. 1 – *Comparaison des vocabulaires*

L'approche Chaos [6] [10] Chaos prend en compte trois aspects de la communication : le réseau de communication où évoluent les utilisateurs, la liste des activités dans lesquelles ils sont engagés au niveau individuel et au niveau du groupe, les liens organisationnels qui lient les membres du groupe en terme d'expérience et de responsabilités.

Ceci permet de tenir compte des aspects pragmatiques et structurant de la communication, auxquels s'ajoute une dimension sémantique par l'analyse de messages en langage naturel semi-structuré. Les types de messages s'inspirent de la théorie des actes de langage de Searle [38]. Chaos distingue trois différents types de rôle (manager, senior, junior) et un rôle d'expertise relativement à un domaine. L'aspect linguistique permet d'acquérir des connaissances sur le langage spécifique des utilisateurs et interpréter le contenu propositionnel des actes de langage.

L'approche Ame [41] Ce modèle, qui est une simulation, se focalise sur les rôles organisationnels, chaque rôle prenant place dans un espace de travail. Il introduit les notions d'activité, de message, d'unité d'information et de règle de fonction. Une base de données contient la description de l'organisation et ses activités en cours et un interpréteur de règles permettant un routage intelligent des messages entre les rôles de l'organisation. L'utilisateur agit en créant et en jouant des rôles dans une zone conceptuelle de travail contenant les ressources associées à chaque rôle particulier.

L'approche Cosmos [13] Ici, chaque utilisateur est engagé dans différentes activités, en assumant un rôle. Pour permettre la configurabilité du modèle, la classe d'objets décrivant les activités inclut la spécification des rôles et des actions ainsi que leur ordonnancement et le type de messages échangés. Cosmos propose la notion de structure de communication comme base d'un système supportant des activités coopératives. Une structure de communication est une description abstraite d'une classe d'activité de communication et inclut la spécification de rôles d'action et leur ordonnancement, ainsi que l'organisation des messages émis et reçus. Une librairie de structures de communication permet d'instancier des activités. Le système conserve des informations sur les activités en cours et les messages échangés par utilisation de bases de données partagées. La fonctionnalité d'un système particulier

est déterminée par les structures de communication qui le composent.

Les approches ci-dessus décrivent les activités, les agents et les rôles, mais les notions d'information sur le produit du processus, la prise de décision, les notions de qualité du processus et de gestion du processus en général ne sont pas prises en compte. Comme le signale Benford [5], les activités n'existent pas de manière isolée car, elles peuvent partager les mêmes ressources, avoir des priorités relatives et supposent donc l'allocation des rôles et des ressources. Le fait de « câbler » à priori des structures d'échanges se rapproche ainsi des modèles de workflow [27].

Chapitre 2

Contrôle des procédures d'affaires

Le contrôle des procédures d'affaires dans une organisation peut prendre ses racines dans la théorie des systèmes, la théorie de la commande, la théorie du contrôle et la cybernétique.

2.1 Formulation du problème de contrôle de procédures d'affaires comme un problème de synthèse de contrôleur

L'enchaînement des procédures d'affaires obéit à trois sous-systèmes à savoir : système opérant (SO), système d'information (SI) et système de pilotage (SP).

- **Le système opérant** est le siège de l'activité productive de l'organisation. Cette activité consiste en une transformation de ressources ou flux primaires.

- **Le système de pilotage** est le siège de l'activité décisionnelle de l'organisation. Cette activité décisionnelle est très large et est assurée par tous les acteurs de l'organisation, à des niveaux divers, depuis les acteurs agissant plutôt dans l'activité

productrice de l'organisation à ceux dirigeant cette dernière. Elle permet la régulation, le pilotage mais aussi l'adaptation de l'organisation à son environnement.

- **Le système d'information** est une représentation de l'activité du système opérant ou du système de pilotage ou les deux à la fois et de ses échanges avec l'environnement, conçu à l'initiative du système de pilotage en fonction des objectifs à atteindre. Ce système d'information est destiné au système de pilotage pour pouvoir connaître et maîtriser le fonctionnement du système opérant et au système opérant lorsque les flux transformés sont de nature « information ».

Dans le cadre de ce mémoire, pour la synthèse de contrôleur des procédures d'affaires, nous substituons le système de pilotage au contrôleur et le système opérant au procédé afin d'intégrer un contrôleur dans l'organisation (voir figure 12).

Le contrôleur supervise le fonctionnement du procédé. Il effectue sa tâche grâce aux variables de contrôle et aux variables de positionnement. Les variables de contrôle contiennent des données qui sont nécessaires pour les prises de décisions du contrôleur. En fonction de chaque décision prise, le contrôleur met à jour les variables de positionnement, ce qui a pour effet de produire un événement dans le procédé ou de prohiber certains événements.

Plus particulièrement, le contrôleur reçoit en entrée un événement et retourne en sortie un vecteur de contrôle approprié qui indique pour chaque événement s'il est permis ou non. Dans le cadre de la théorie du contrôle des procédures d'affaires, le problème consiste à exprimer des propriétés sur l'organisation, considérées comme des règles de gestion (contraintes), et à guider son comportement grâce à des actions de contrôle de sorte que l'ensemble des suites d'événements de l'organisation contrôlé possède les propriétés désirées. Le problème consiste donc à construire ou à dériver un contrôleur qui agit sur le procédé pour restreindre son comportement aux suites d'événements permises.

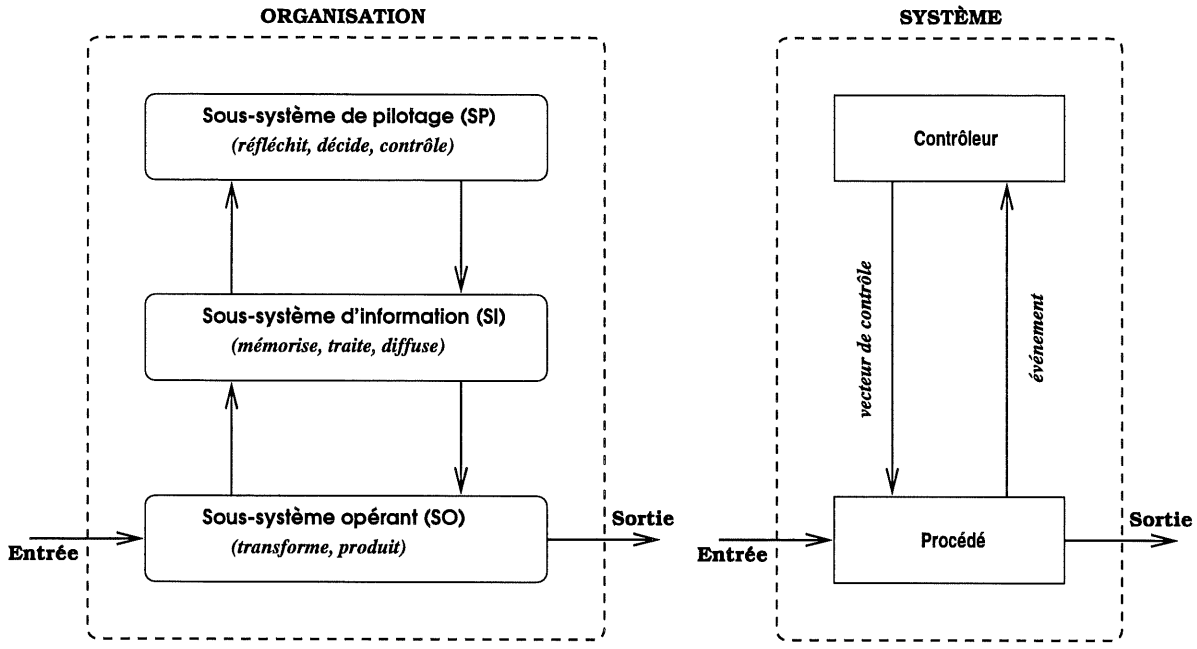

FIG. 12 – *Modèle adopté pour le contrôle de procédures d'affaires*

2.2 Formalismes utilisés dans la description des procédures d'affaires

Le formalisme de Merise pour décrire la dynamique des procédures d'affaires s'appuie sur quatres concepts à savoir : événement, résultat, opération et synchronisation. Un ensemble d'événements déclenche, après combinaison par une synchronisation, une opération qui produit un ensemble de résultats en sortie.

La dynamique des procédures d'affaires dans OMT représente l'évolution du système dans le temps en termes d'événements et d'états.

Le comportement dynamique des procédures d'affaires dans OSSAD est représenté à l'aide des modèles : abstrait, descriptif et perscriptif en termes de fonctions, rôles et documents.

Le comportement dynamique des procédures d'affaires dans CORIG [9] et ATLAS [31] est représenté en termes de **procédures fonctionnelles** dont chacune est activée par un **événement** et produit un ou plusieurs **résultats**. Malgré des développements orientés vers la prise en compte du temps réel, ces deux méthodes ignorent trop souvent les interactions nécessaires entre les différents traitements. En effet, l'utilisation commune d'un ensemble de données et le partage d'un processeur ou d'un réseau informatique peuvent provoquer des impasses entre les différentes opérations.

Les représentations graphiques des modèles conceptuels et des modèles organisationnels des traitements de la méthode Merise sont proches des réseaux de Petri. De plus, pour la simulation de ces modèles ou pour la synthèse de contrôleurs de procédures d'affaires, il est nécessaire de disposer de langages formels de représentation tels que les réseaux de Petri ou les automates.

2.2.1 Réseau de Petri

Les réseaux de Petri peuvent être utilisés comme un outil de représentation, de validation et de vérification des procédures d'affaires [24]. Les réseaux de Petri (RdP) permettent de tenir compte du comportement dynamique des systèmes et ils sont utilisés dans des modèles tels que Merise. Les RdP fournissent également un formalisme graphique (plus précis et formel que les diagrammes de flux par exemple), donnant un outil de communication fin entre les concepteurs et les utilisateurs. Nous rappelons ici les principaux concepts propres aux réseaux de Petri. Cette présentation est une adaptation des références [30] [39].

Un réseau de Petri est un quadruplet $R = \langle P, T, I, O \rangle$ où,

- P est un ensemble fini de places,
- T est un ensemble fini de transitions,
- $I : P \times T \to \mathbb{Z}$ est l'application incident avant (places précédentes),
- $O : P \times T \to \mathbb{Z}$ est l'application incident arrière (places suivantes).

Réseau marqué

Un réseau marqué est le couple $N = \langle R, M_0 \rangle$ où,

- R est un réseau de Petri,
- M_0 est le marquage initial, c'est-à-dire une application $P \to \mathbb{Z}$.

Un marquage de R est aussi une application $P \to \mathbb{Z}$ et $M(p)$ est le nombre de jetons contenus dans la place p.

Graphe associé et notation matricielle

À un réseau de Petri, on peut associer un graphe qui possède deux types de nœud (les places et les transitions). Un arc relie une place p à une transition t si et seulement si $I(p,t) \neq 0$. Les valeurs non nulles des applications I et O sont associées aux arcs comme étiquettes (par défaut, on prend la valeur 1). Un exemple de graphe associé à un réseau de Petri est donné par la figure 13.

Le marquage M_0 peut être représenté par un vecteur ayant pour dimension le nombre de places. Les applications I et O peuvent être représentées sous une forme matricielle. Le nombre de lignes de chaque matrice est égal au nombre de places et le nombre de colonnes est égal au nombre de transitions. Dans ce contexte, nous utilisons la notation $C = O - I$.

On note $I(.,t)$, $O(.,t)$ et $C(.,t)$ les colonnes des matrices I, O et C associées à une transition t. Ce sont des vecteurs ayant pour dimension le nombre de places.

Considérons l'exemple de la figure 13. Ce graphe est celui du réseau de Petri, où

$$\begin{aligned} P &= \{p_1, p_2, p_3\} \\ T &= \{a, b, c, d\} \end{aligned}$$

$$I = \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \left[\begin{array}{cccc} 0 & 1 & 0 & 0 \\ 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array} \begin{array}{c} p_1 \\ p_2 \\ p_3 \end{array}$$

$$O = \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 \end{array}\right] \end{array} \begin{array}{c} p_1 \\ p_2 \\ p_3 \end{array}$$

On a alors :

$$C = \left[\begin{array}{cccc} 1 & -1 & 0 & 0 \\ -1 & 1 & -3 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right] \begin{array}{c} p_1 \\ p_2 \\ p_3 \end{array}$$

Le marquage initiale est :

$$M_0 = \left[\begin{array}{c} 0 \\ 3 \\ 0 \end{array}\right]$$

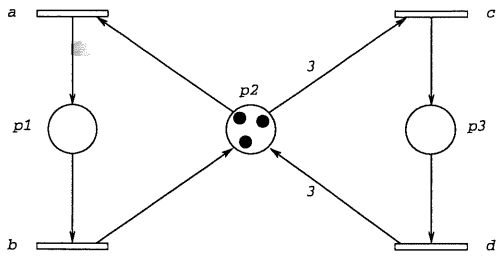

FIG. 13 – *Exemple d'un réseau de Petri*

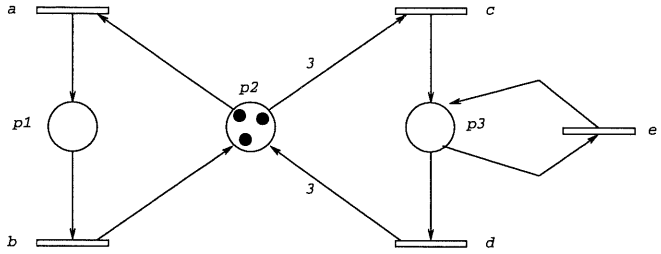

FIG. 14 – *Exemple de réseau de Petri non pur*

Réseau de Petri pur

Un réseau de Petri R est pur si et seulement si :

$$\forall p \in P \text{ et } \forall t \in T : I(p,t) \times O(p,t) = 0$$

Le graphe associé à un réseau de Petri pur ne comprend aucune boucle élémentaire, c'est-à-dire aucune transition ayant la même place en entrée et en sortie. Par exemple, le réseau de Petri de la figure 13 est pur. Par contre, celui de la figure 14 ne l'est pas.

Transition franchissable

Une transition t est franchissable si et seulement si pour tout $p \in P$, $M(p) \geq I(p,t)$ On peut exprimer que t est franchissable par les notations

$$M \geq I(.,t),\ M(t> \text{ ou } M \xrightarrow{t}$$

Par exemple, dans le réseau de Petri de la figure 13, les transitions a et c sont franchissables, car

$$I(.,a) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad I(.,c) = \begin{bmatrix} 0 \\ 3 \\ 0 \end{bmatrix}$$

$$M_0 \geq I(.,a) \text{ et } M_0 = I(.,c)$$

Franchissement d'une transition

Si t est franchissable pour le marquage M, le franchissement de t donne un nouveau marquage M' tel que pour tout $p \in P$:

$$M'(p) = M(p) - I(p,t) + O(p,t)$$

On utilise également les notations

$$M' = M - I(.,t) + O(.,t),\ M(t > M' \text{ ou } M \xrightarrow{t} M'$$

Par exemple dans le réseau de la figure 13, après le franchissement de a à partir du marquage initial M_0, on obtient le marquage suivant :

$$\begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Conflit et parallélisme

Conflit structurel Deux transitions t_1 et t_2 sont en conflit structurel si et seulement si elles ont au moins une place d'entrée en commun :

$$\exists p : I(p,t_1) \times I(p,t_2) \neq 0$$

Conflit effectif Deux transitions t_1 et t_2 sont en conflit effectif pour un marquage M si et seulement si t_1 et t_2 sont en conflit structurel, et

$$M \geq I(.,t_1) \text{ et } M \geq I(.,t_2)$$

Parallélisme structurel Deux transitions t_1 et t_2 sont structurellement parallèles si $(I(.,t_1))^T I(.,t_2) = 0$. Elles n'ont donc aucune place d'entrée commune (le produit de leur vecteur correspondant à I est nul).

Parallélisme effectif Deux transitions t_1 et t_2 sont parallèles pour un marquage donné M si et seulement si elles sont structurellement parallèles, et

$$M \geq I(.,t_1) \text{ et } M \geq I(.,t_2)$$

Ainsi, dans le réseau de Petri de la figure 13, les transitions a et c sont en conflit structurel puisque $I(p_2,a) \times I(p_2,c) = 3$. Les transitions b et d sont structurellement parallèles. En effet,

$$I(.,b) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \text{ et } I(.,d) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Pour le marquage initial M_0, les transitions a et c sont en conflit effectif. Si on considère le marquage

$$M = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

alors les transitions b et d sont effectivement parallèles (elles peuvent être franchies indépendamment l'une de l'autre).

Séquence de franchissement

Si $M_1 \xrightarrow{t_a} M_2$ et $M_2 \xrightarrow{t_b} M_3$, on dit que la séquence $t_a t_b$ est franchissable à partir de M_1, ce que l'on note $M_1 \xrightarrow{t_a t_b} M_3$ ou encore $M_1(t_a t_b) > M_3$.

Soit $\bar{s}$ un vecteur dont sa dimension est égale au nombre de transitions du réseau de Petri. La composante $\bar{s}(t)$ de ce vecteur donne le nombre d'occurrences de la transition t dans une séquence de franchissement s. Ce vecteur est appelé vecteur caractéristique de s. En considérant le réseau de Petri de la figure 13, on a, par exemple, pour la séquence $s = aab$:

$$\begin{bmatrix} 0 \\ 3 \\ 0 \end{bmatrix} \xrightarrow{t_a} \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \xrightarrow{t_a} \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix} \xrightarrow{t_b} \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}$$

et

$$\overline{aab} = \begin{bmatrix} 2 \\ 1 \\ 0 \\ 0 \end{bmatrix} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

Les évolutions du marquage d'un réseau de Petri sont alors données par l'équation

$$M' = M - I\bar{s} + O\bar{s}$$

que l'on peut également écrire :

$$M' = M + C\bar{s} \text{ avec } M \geq 0, \bar{s} \geq 0$$

Cette équation est appelée l'équation fondamentale d'un réseau de Petri. Il n'est pas suffisant de trouver un vecteur caractéristique $\bar{s}$ vérifiant la dernière équation pour être sûr qu'il existe une séquence s effectivement franchissable du marquage M vers le marquage M'. En effet, il faut que le marquage de départ soit tel que les transitions seront effectivement franchissables pour chaque marquage intermédiaire. Considérons par exemple le réseau de Petri de la figure 15. Si

$$M = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad C = \begin{bmatrix} -1 & -1 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \qquad \overline{abcd} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

alors $M' = M + C\overline{abcd}$, mais en fait on peut vérifier que la séquence $s = abcd$ n'est pas franchissable et donc on ne peut pas écrire $M \xrightarrow{abcd} M'$.

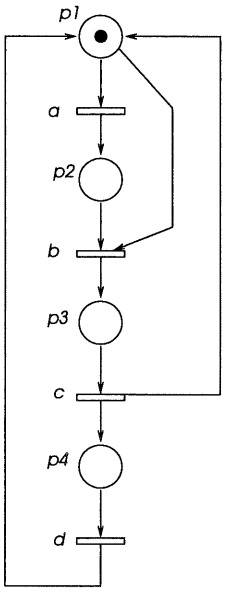

FIG. 15 – *Exemple de réseau de Petri avec la séquence abcd non franchissable*

Ensemble des marquages accessibles

L'ensemble des marquages accessibles $A(R; M_0)$ d'un réseau de Petri marqué est l'ensemble des marquages que l'on peut atteindre à partir du marquage initial M_0 par une séquence de franchissement.

$$A(R; M_0) = \{M_i : \exists s \in T^* \text{ tel que } M_0 \xrightarrow{s} M_i\}$$

On peut, lorsque cet ensemble est fini, le représenter sous la forme d'un graphe $GA(R; M_0)$. Ce graphe a pour ensemble de sommets l'ensemble des marquages accessibles $A(R; M_0)$, un arc relie deux sommets M_i et M_j s'il existe une transition t franchissable permettant de passer d'un marquage à un autre : $M_i \xrightarrow{t} M_j$.

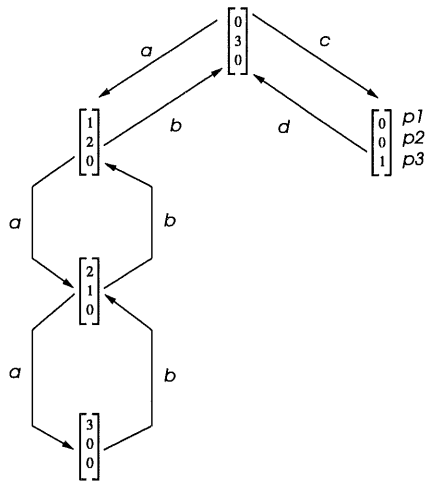

FIG. 16 – *Exemple de graphe des marquages accessibles*

En général, les arcs du graphe sont étiquetés par les transitions correspondantes. C'est en fait l'automate équivalent au réseau de Petri. Il faut toutefois bien remarquer que, la notion de processus ayant disparu, on ne peut plus faire la distinction entre les transitions parallèles et les transitions en conflit pour un marquage donné. La figure 16 représente le graphe des marquages accessibles pour le réseau de Petri de la figure 13.

Vues d'un réseau de Petri

Jusqu'à maintenant, un réseau de Petri pouvait être vu sous l'un des deux aspects suivants :

- un graphe avec deux types de sommets et un comportement dynamique,
- un ensemble de matrices d'entiers non négatifs dont le comportement dynamique est décrit par un système linéaire (équation fondamentale).

Il peut également être considéré comme un système de production de connaissances (système de déduction), appelé système de règles et fondé sur une représentation des connaissances de la forme :

$$CONDITION \Longrightarrow ACTION$$

Système de règles

Un système de règles ou système de production de connaissances est formé :

- d'une base de faits (contexte des faits connus augmenté des règles permettant de déduire de nouveaux faits),
- d'un mécanisme d'inférence (c'est-à-dire de déduction).

Un mécanisme d'inférence assez élémentaire consiste à ranger l'ensemble des règles dans une liste et à parcourir cette liste séquentiellement. Dès que la condition d'une règle est vraie dans le contexte courant (faits vérifiés au moment de d'inférence), la règle est appliquée. Si aucune règle n'est applicable ou si un fait terminal (conclusion) devient vrai, le mécanisme d'inférence s'arrête.

Le plus souvent, dans un contexte donné, plusieurs règles peuvent être applicables et le résultat de la déduction peut être différent suivant que l'on choisit d'appliquer d'abord telle ou telle règle ou de les appliquer toutes simultanément. Le mécanisme élémentaire choisit la première règle rencontrée sans même détecter que d'autres possibilités existent.

Une situation où plusieurs règles sont applicables est appelée situation de conflit. La résolution des conflits, encore appelée le contrôle, caractérise en fait le mécanisme d'inférence qui peut alors devenir très complexe.

Dans le cas d'un réseau de Petri :

- les matrices I et O, exploitées colonne par colonne, (c'est-à-dire l'ensemble des transitions avec leurs règles de franchissement) sont la base de règles,

- le marquage (initial) est le contexte (initial),

- la résolution des conflits s'apparente aux notions de transitions parallèles (l'ordre des franchissements est indifférent) et de transitions en conflit (le résultat dépendra de l'ordre des franchissements) ; si deux transitions sont en conflit effectif, on en franchit une choisie de façon aléatoire et l'autre n'est pas franchie.

Grammaire

Un réseau de Petri est toutefois un système de règles bien particulier. La base de faits n'est pas générale, car elle est construite uniquement à partir de la notion de marquage des places, c'est-à-dire de la valeur d'un ensemble fini de compteurs. Il est possible de décrire cela à l'aide de mots construits sur un alphabet. L'alphabet est l'ensemble des identificateurs des places.

Par exemple, un marquage pour lequel la place p_1 contient un jeton et la place p_3 contient deux jetons s'écrit $p_1p_3p_3$ ou $p_1p_3^2$.

On peut définir une application μ, qui à tout marquage M et, plus généralement, à tout vecteur défini sur l'ensemble des places (les colonnes de I et de O), associe un mot μ(M) de P^* (P^* est l'ensemble des suites finies d'éléments de P et incluant aussi l'élément vide λ). La classe particulière de système de règles correspondant à un réseau de Petri s'appelle une grammaire (système de réécriture de mots).

La grammaire $S\langle P; Q\rangle$ associée au réseau $R = \langle P, T, I, O\rangle$ est définie par :

- son vocabulaire P,

- l'ensemble Q des règles de réécriture, où on retrouve pour chaque t_i, la règle : $\mu(I(., t_i)) \rightarrow \mu(O(., t_i))$.

À un marquage initial M_0 du réseau correspond un axiome $A = \mu(M_0)$ de la grammaire, à partir duquel on peut dériver de nouveaux mots.

Quant on considère les séquences de franchissement, si T est considéré comme un alphabet, T^* représente l'ensemble des séquences de franchissement pour lesquels il est important de tenir compte de l'ordre des éléments. Par contre, les marquages sont des éléments de P^* pour lequel l'ordre n'a aucune importance.

Considérons le réseau de Petri de la figure 13. Interprété comme une grammaire, nous aurons $P = \{p_1, p_2, p_3\}$ et

$$Q = \begin{cases} a : p_2 \rightarrow p_1 \\ b : p_1 \rightarrow p_2 \\ c : p_2^3 \rightarrow p_3 \\ d : p_3 \rightarrow p_2^3 \end{cases}$$

$$A(R; M_0) = \{p_2^3, p_2^2 p_1, p_2 p_1^2, p_1^3, p_3\}$$

Cette interprétation est importante, car elle permet des notations abrégées et surtout de faire le pont entre les réseaux de Petri et les techniques d'intelligence artificielle.

Un réseau de Petri peut donc être vu de trois façons différentes, chacune présentant certains avantages et certains inconvénients.

Le graphe, lorsqu'il est de taille raisonnable, produit une description facilement transmissible permettant d'expliquer clairement certains mécanismes de synchronisation. Les jetons visualisent les objets et les circuits correspondent à des processus séquentiels répétitifs.

La représentation matricielle, résumée par l'équation fondamentale, caractérise un sur-ensemble (car s doit être une séquence effectivement franchissable) de l'ensemble des marquages accessibles par un système d'équations linéaires.

Enfin, nous pouvons considérer un réseau de Petri comme un système de règles particulier. Cet aspect est intéressant quant on veut effectuer une mise en œuvre dans un contexte plus général, mais orienté vers les techniques de l'intelligence artificielle.

« Bonnes » propriétés

Dans cette section, nous définissons un certain nombre de propriétés concernant les réseaux de Petri marqués (elles dépendent donc du marquage initial). Elles sont en général regroupées sous le nom générique de « bonnes » propriétés. Leur définition implique des considérations sur l'ensemble des marquages accessibles à partir du marquage initial.

K-borné

Place k-bornée et binaire Une place p d'un réseau marqué N est k-bornée si et seulement si $\forall M' \in A(R; M_0)$, $M'(p) \leq k$. Si $k = 1$, on dit que la place est binaire (*safe*).

Si on considère le réseau de Petri de la figure 13, on voit que pour le marquage initial $M_0 = p_2^3$, la place p_3 est binaire alors que les places p_1 et p_2 sont 3-bornées.

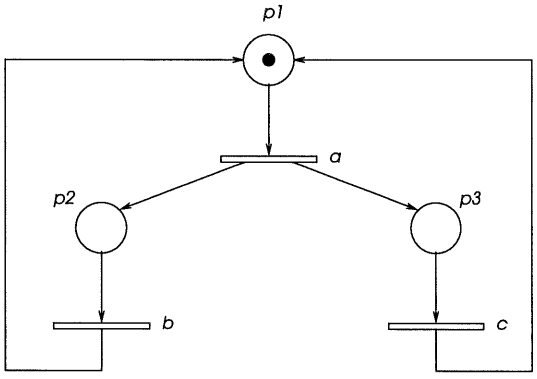

FIG. 17 – *Réseau de Petri non borné*

Réseau de Petri k-borné et binaire Un réseau de Petri marqué N est k-borné (*bounded*) si et seulement si toutes ses places sont k-bornées. Un réseau Petri marqué N est binaire (*safe*) si et seulement si toutes ses places sont binaires.

Exemples Le réseau de Petri de la figure 13 est 3-borné pour le marquage $M_0 = p_2^3$. Si son marquage initial était $M_0 = p_2$, il serait binaire (les transitions c et d ne seraient toutefois jamais franchies).

Considérons par contre le réseau de Petri de la figure 17 pour le marquage initial $M_0 = p_1$. Chaque fois que l'on franchit la séquence $s = ab$, on rajoute un jeton dans la place p_3. Cette place est donc non bornée et le réseau l'est également (les places p_1 et p_2 sont également non bornées).

Le concept de réseau binaire présente l'intérêt suivant. Si les places représentent des conditions logiques, la présence de plus d'un jeton dans une place signifie qu'une incohérence s'est glissée dans le modèle. Généralement, il s'agit d'une condition logique mise à 1 lors d'un cycle de fonctionnement, qui n'a pas été utilisée (mise à 0) et qui est remise à 1 au cycle suivant.

Le concept de réseau borné correspond au fait qu'un système physique est toujours limité. Toutefois, on peut être amené à utiliser des réseaux de Petri non bornés lorsque l'on veut évaluer les performances d'un système indépendamment des bornes de ses éléments de stockage intermédiaires.

Un exemple de réseau de Petri non borné décrivant un mécanisme classique est donné par la figure 18. Si au franchissement de la transition a on associe l'ouverture d'une parenthèse (parenthèse gauche) et à celui de la transition b la fermeture d'une parenthèse (parenthèse droite), les séquences de franchissement de transition qui à partir du marquage initial M_0 ($M_0(p_1) = 0$) conduisent à ce même marquage M_0 décrivent toutes les expressions parenthèsées licites. Il est clair que ce réseau n'est pas borné, car on peut toujours ouvrir une parenthèse (franchir a) et donc rajouter un jeton dans la place p_1.

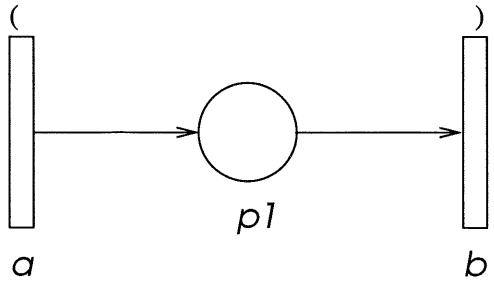

FIG. 18 – *Exemple du réseau de Petri « parenthèses »*

Vivant

Transition quasi vivante Une transition t d'un réseau de Petri marqué N avec $N = \langle R, M_0 \rangle$ est quasi vivante si et seulement s'il existe une séquence de franchissement s telle que $M_0(s > M'$ et $M'(t >$. Ce que l'on peut également écrire $M_0 \xrightarrow{st}$.

Transition vivante Une transition t d'un réseau de Petri marqué N avec $N = \langle R, M_0 \rangle$ est vivante si et seulement si $\forall M' \in A(R; M_0) : \exists s \in T^*$ tel que $M' \xrightarrow{st}$.

Exemple Considérons le réseau de la figure 19. Le graphe des marquages associé au marquage initial $M_0 = p_3 p_4$ est donné par la figure 20. Il est clair que la transition d est quasi vivante (elle peut être franchie une fois), mais non vivante (elle ne peut être franchie qu'une fois, car on ne peut plus sortir de la composante fortement connexe formée par les marquages p_1, p_2, p_3 et p_4).

Par contre les transitions a, b, c, e et f sont vivantes, car elles figurent dans une composante fortement connexe et on pourra donc toujours trouver une séquence de franchissement les contenant.

Une transition peut, sans être vivante, apparaître une infinité de fois dans des séquences infinies de franchissement de transition. C'est le cas de la transition g dans la figure 21. Elle peut être franchie autant de fois que l'on veut avant

le franchissement de d, mais après elle ne peut plus être franchie.

Réseau de Petri marqué vivant Un réseau de Petri marqué $N = \langle R, M_0 \rangle$ est vivant si et seulement si toutes ses transitions sont vivantes.

- On considère un réseau de Petri pour un marquage initial donné (réseau marqué).
- Un réseau de Petri vivant garantit qu'aucun blocage ne peut être provoqué par la structure du réseau de Petri. Par contre, il ne prouve pas l'absence d'éventuels blocages provoqués par une mauvaise interaction entre le réseau de Petri et son environnement.
- Un réseau de Petri vivant garantit également l'absence de partie morte (jamais atteinte).

Par exemple, le réseau de Petri de la figure 13 est vivant pour le marquage initial donné, par contre celui de la figure 19 ne l'est pas. Un réseau de Petri peut très bien être non borné et vivant ; c'est par exemple le cas du réseau de Petri de la figure 18 (la séquence ab est toujours franchissable quel que soit le marquage).

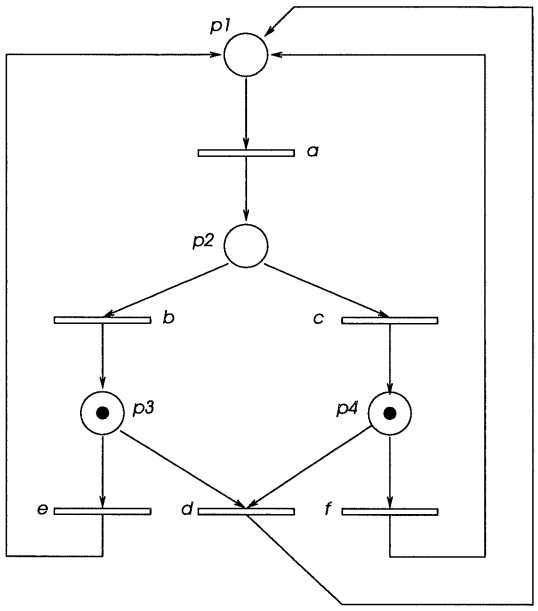

FIG. 19 – *Exemple de réseau de Petri avec transition quasi vivante et non vivante*

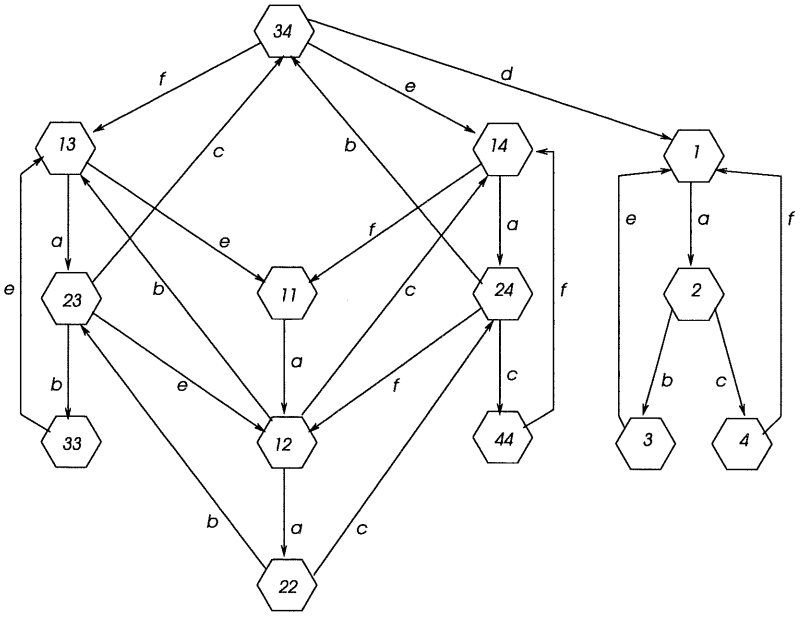

FIG. 20 – *Graphe des marquages (transition quasi vivante)*

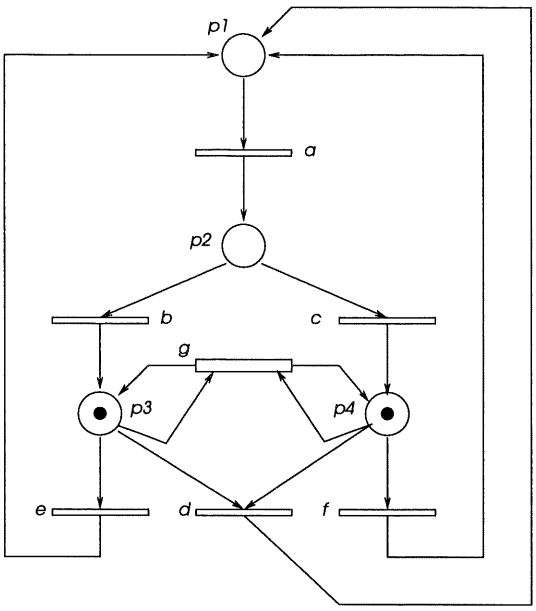

FIG. 21 – *Exemple de réseau de Petri avec transition quasi vivante et séquence infinie*

Réinitialisable

Réseau de Petri marqué réinitialisable Un réseau de Petri marqué $N = \langle R, M_0 \rangle$ est réinitialisable (propre) si et seulement si son graphe des marquages accessibles $GA(R; M_0)$ est fortement connexe :

$$\forall M' \in A(R; M_0), \exists s \in T^* \text{ tel que } M' \xrightarrow{s} M_0$$

La plupart des systèmes ont des fonctionnements répétitifs et donc les réseaux de Petri utilisés seront réinitialisables.

Exemple Considérons le réseau de Petri marqué de la figure 22 dont le graphe des marquages accessibles est donné par la figure 23. Il est non réinitialisable puisqu'il n'existe aucune séquence permettant de revenir au marquage initial $M_0 = p_1 p_4$ après le franchissement de la transition a.

Par contre il est intéressant de noter qu'il est vivant, car à l'intérieur de la composante fortement connexe formée par les marquages $p_2 p_4$, $p_1 p_3$ et $p_2 p_3$, il

est toujours possible de franchir toutes les transitions du réseau de Petri soit a, b et c. En conséquence, si on considère le réseau de la figure 22 pour l'un des marquages initiaux suivants $M_0 = p_2p_4$, $M_0 = p_1p_3$ ou $M_0 = p_2p_3$, il est à la fois vivant et réinitialisable. Inversement, le réseau de Petri de la figure 19 est réinitialisable pour le marquage $M_0 = p_1$ sans être vivant puisque la transition d ne peut jamais être franchie.

Il est important de bien remarquer que les propriétés que nous venons de définir sont fortement liées au marquage initial. Considérons par exemple le réseau de Petri de la figure 24. Pour le marquage $M_0 = p_1p_3$, il est binaire, vivant et réinitialisable. Si on ajoute un jeton dans la place p_4, alors il cesse d'être borné (voir la séquence $abab$). Enfin, pour le marquage $M_0 = p_1$, le réseau est binaire et non vivant (aucune transition franchissable).

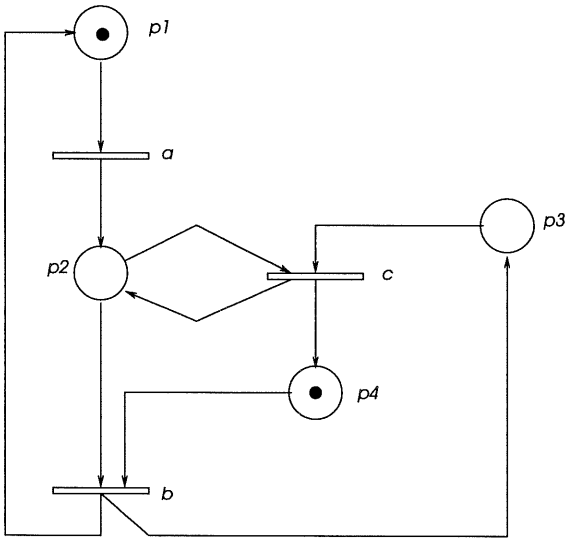

FIG. 22 – *Exemple de réseau de Petri non réinitialisable*

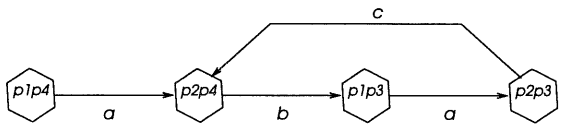

FIG. 23 – *Graphe des marquages accessibles du réseau de Petri de la figure 22*

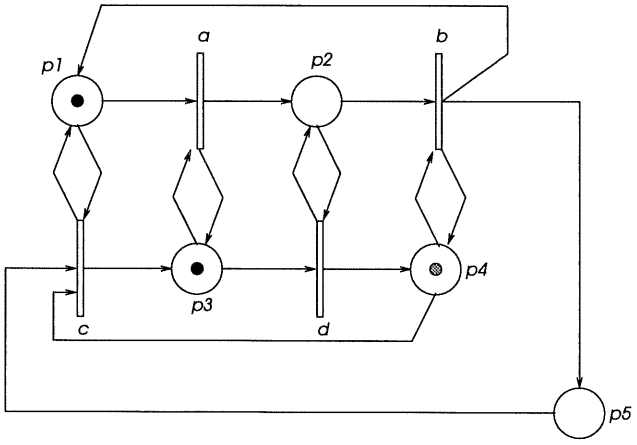

FIG. 24 – *Réseau de Petri illustrant certaines propriétés*

2.2.2 Les automates

Nous pouvons aussi modéliser le comportement dynamique de procédure d'affaires à l'aide des automates. Chaque automate met en évidence les séquences d'états et d'événements permises dans une organisation pour un acteur de procédure d'affaires.

Définition

Un automate G est un 5-uplet $(X, \Sigma, \delta, x_0, X_m)$, où X est l'ensemble des états de l'automate (si X est fini, nous avons des automates finis), Σ est l'alphabet, δ est la fonction de transition, $x_0 \in X$ est l'état initial et $X_m \subseteq X$ est l'ensemble des états finaux (on dit aussi marqués).

La fonction de transition peut être représentée par une table ou un diagramme de transitions d'états. On désigne par Σ^* l'ensemble des chaînes finies sur l'alphabet Σ et par ϵ la chaîne de longueur vide. Il y a différents types d'automates selon la signature de δ :

- automates non déterministes avec transitions spontanées ($\delta : X \times (\Sigma \cup \epsilon) \to 2^X$)
- automates non déterministes ($\delta : X \times \Sigma \to 2^X$)
- automates déterministes ($\delta : X \times \Sigma \to X$)

Langages associés à un automate

La $\epsilon - fermeture$ d'un état $x \in X$, notée $\epsilon^*_G(x) \subseteq X$ est définie par :

$$x \in \epsilon^*_G(x)$$

$$x' \in \epsilon^*_G(x) \Rightarrow \delta(x', \epsilon) \subseteq \epsilon^*_G(x)$$

L'extension de la fonction de transition δ aux chaînes de Σ^*, $\delta : X \times \Sigma^* \rightarrow 2^X$, est définie par :

$$\delta(x, s\sigma) = \bigcup_{x' \in \delta(x,s)} \{ \bigcup_{x'' \in \delta(x',\sigma)} \epsilon^*_G(x'') \}$$

Le langage généré par G, noté $L(G)$, et le langage marqué de G, noté $L_m(G)$, sont définis comme suit :

$$\begin{aligned} L(G) &= \{s \in \Sigma^* \mid \delta(x_0, s) \neq \emptyset\} \\ L_m(G) &= \{s \in L(G) \mid \delta(x_0, s) \cap X_m \neq \emptyset\} \end{aligned}$$

Accessibilité (pour un automate déterministe)

Soit $Re_G(x) = \{x' \in X \mid \exists s \in \Sigma^* \text{ tel que } \delta(x, s) = x'\}$ l'ensemble des états accessibles à partir de $x \in X$. Un état x est co-accessible s'il existe une chaîne $s \in \Sigma^*$ tel que $\delta(x, s)$ est définie et $\delta(x, s) \in X_m$, c'est-à-dire que pour tout $x \in X, Re_G(x) \cap X_m \neq \emptyset$. Un automate est accessible si tous ses états sont accessibles à partir de l'état initial, c'est-à-dire que $Re_G(x_0) = X$. Il est co-accessible, dit aussi non bloquant, si tous ses états sont co-accessibles, c'est-à-dire $L(G) = \overline{L_m(G)}$ (la fermeture de $L_m(G)$, c'est-à-dire l'ensemble de toutes les chaînes qui sont des préfixes d'au moins une chaîne de $L(G)$). G est soigné s'il est à la fois accessible et co-accessible.

La composante accessible de G, notée $Ac(G) = (X_{ac}, \Sigma, \delta_{ac}, x_0, X_{ac,m})$, est un automate qui possède l'ensemble des états accessibles de G, c'est-à-dire :

- $X_{ac} = Re_G(x_0)$
- $\delta_{ac} = \delta \mid_{X_{ac} \times \Sigma}$
- $X_{ac,m} = X_{ac} \cap X_m$

Composition synchrone

Soit $G_1 = (X_1, \Sigma_1, \delta_1, x_{0,1}, X_{m,1})$ et $G_2 = (X_2, \Sigma_2, \delta_2, x_{0,2}, X_{m,2})$. La composition synchrone de G_1 et G_2, noté $G_1 \parallel G_2$ est définie comme suit :

- $G_1 \parallel G_2 = (X, \Sigma, \delta, x_0, X_m)$
- $X = X_1 \times X_2$
- $\Sigma = \Sigma_1 \cup \Sigma_2$
- $x_0 = (x_{0,1}, x_{0,2})$
- $X_m = X_{m,1} \times X_{m,2}$
- pour tout $x = (x_1, x_2) \in X, \sigma \in \Sigma$:

$$\delta(x,\sigma) = \begin{cases} (\delta_1(x_1,\sigma), \delta_2(x_2,\sigma)) & \text{si } \delta_1(x_1,\sigma) \text{ et } \delta_2(x_2,\sigma) \text{ sont définies} \\ & \text{et } \sigma \in \Sigma_1 \cap \Sigma_2 \\ (\delta_1(x_1,\sigma), x_2) & \text{si } \delta_1(x_1,\sigma) \text{ est définie }, \sigma \in \Sigma_1 - \Sigma_2 \\ (x_1, \delta_2(x_2,\sigma)) & \text{si } \delta_2(x_2,\sigma) \text{ est définie }, \sigma \in \Sigma_2 - \Sigma_1 \\ \text{non définie} & \text{autrement} \end{cases}$$

Nous avons

$$\begin{aligned} L(G_1 \parallel G_2) &= \{s \in \Sigma^* \mid s \uparrow \Sigma_1 \in L(G_1) \text{ et } s \uparrow \Sigma_2 \in L(G_2)\} \\ L_m(G_1 \parallel G_2) &= \{s \in \Sigma^* \mid s \uparrow \Sigma_1 \in L_m(G_1) \text{ et } s \uparrow \Sigma_2 \in L_m(G_2)\} \end{aligned}$$

où $\uparrow$ est la projection d'une chaîne par rapport à un alphabet.

Si $\Sigma_1 \cap \Sigma_2 = \emptyset$, alors $G_1 \parallel G_2$ est la composition intercalée (shuffle) de G_1 et G_2. Si $\Sigma_1 = \Sigma_2$, alors on obtient le produit de G_1 et G_2, noté $G_1 \times G_2$. En particulier,

$$L(G_1 \times G2) = L(G_1) \cap L(G_2)$$

$$L_m(G_1 \times G2) = L_m(G_1) \cap L_m(G_2)$$

2.3 Processus de génération de contrôleurs

Dans cette section, nous décrivons le processus de génération de contrôleurs à l'aide des automates ou des réseaux de Petri.

Selon Zhou et DiCesare [49] la méthode de modélisation est un facteur clé dans les applications des réseaux de Petri. Les étapes de construction d'un modèle de réseaux de Petri sont :

- identifier les activités et les ressources nécessaires,
- structurer les activités concourant à l'élaboration d'un résultat,
- créer pour chaque activité une place et l'étiqueter,
- spécifier le marquage initial pour le système.

Dans le cadre de ce mémoire, pour la modélisation et la synthèse de contrôleurs de procédures d'affaires, nous avons respecté les étapes suivantes :

- établir le modèle organisationnel des traitements de la méthode Merise pour chaque procédure d'affaires,
- traduire le modèle organisationnel des traitements en réseaux de Petri,
- injecter des places d'entrée externes [45] au réseau de Petri obtenu à l'étape précédente.

Chaque procédure d'affaires est obtenue en regroupant, d'une part, les règles de déclenchement et d'émission et, d'autre part, les actions de création, de consultation et de mise à jour des entités (famille d'objets) mémorisées. Du point de vue dynamique, ces mêmes entités vont être décrites par leurs sollicitations ou par les réactions qu'elles déclenchent de la part du système d'information, donc par les traitements dont elles sont les causes ou les conséquences. Ceci sera fait en termes d'événements, de synchronisation

et d'opérations. Les événements peuvent être séquentiels ou simultanés. Cette cœxistence dans le temps est précisée par une synchronisation qui décrit une condition de franchissement. Ensuite, il est nécessaire d'expliciter les règles de déclenchement ainsi que les synchronisations des tâches effectuées par les différents acteurs. On obtient ainsi les différents événements associés aux différentes tâches. À partir de la procédure d'affaires, en privilégiant le point de vue des acteurs, on définit une procédure acteur ayant pour but de mettre en évidence la rôle de chacun d'eux.

Le comportement global de l'organisation est formé par l'ensemble des procédures d'affaires et des règles de gestion (voir figure 25). Les procédures d'affaires et les règles sont généralement modélisées à l'aide du MOT de la méthode Merise. Pour synthétiser des contrôleurs à partir des comportements dynamiques des procédures d'affaires et des règles de gestion, nous traduisons le MOT en réseau de Petri et nous injectons des places d'entrée externes au réseau de Petri obtenu.

La modélisation du comportement dynamique de procédures d'affaires à l'aide d'automates consiste en un ensemble de diagrammes de transitions d'états, généralement un diagramme par entité, qui s'exécutent concurremment et dont les changements d'état sont synchronisés. L'état d'un objet est défini par les valeurs de ses attributs et de ses liens. La description comportementale d'un objet doit spécifier ce que fait l'objet en réponse aux événements. Une activité est une opération qui nécessite un certain temps d'exécution. Elle est associée à un état.

Nous modélisons le comportement dynamique de l'ensemble des objets de chaque procédure acteur ainsi que chaque règle de gestion à l'aide d'un automate. L'automate décrivant le comportement dynamique de la procédure d'affaires est alors obtenu par composition synchrone des automates propres aux objets des procédures acteurs. Enfin, l'automate décrivant le comportement global de l'organisation est obtenu par composition synchrone des automates propres aux procédures d'affaires. Le langage légal est aussi représenté par un automate, celui obtenu à partir de l'intersection des automates qui

modélisent les règles de gestion. À l'aide de l'outil SUCSEDES [32], nous générons un contrôleur qui permet de satisfaire les règles de gestion.

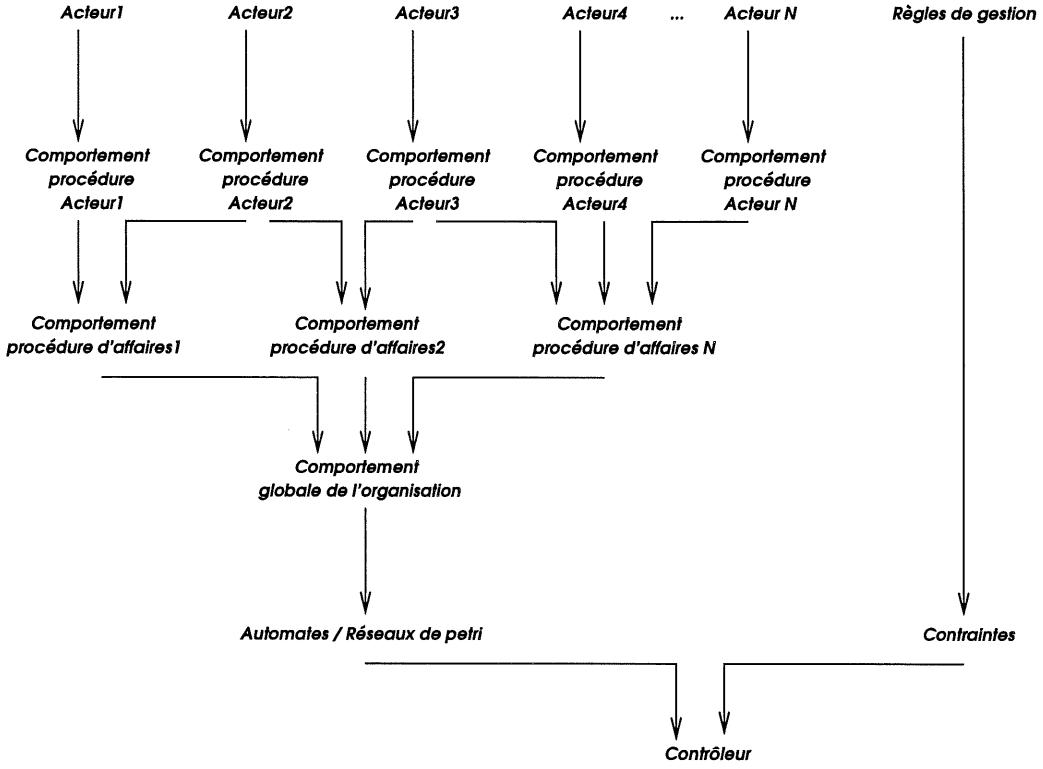

FIG. 25 – *Processus de synthèse du contrôleur*

2.3.1 Approche par automates

Nous rappelons brièvement quelques éléments de la théorie de contrôle des systèmes à événements discrets proposée par Ramadge et Wonham [35] ainsi que le problème de base de la synthèse de contrôleurs.

Le modèle de la figure 26 est très utilisé pour l'étude théorique des systèmes à événements discrets. Un système à événements discrets (ou procédé) est modélisé par un automate fini déterministe $G = (Q, \Sigma, \delta, q_0, Q_m)$. Un état marqué représente généralement une tâche complète.

L'ensemble des événements est divisé en deux sous-ensembles, $\Sigma = \Sigma_c \cup \Sigma_u$, où Σ_c est l'ensemble des événements contrôlables et Σ_u est l'ensemble des événements incontrôlables. Les événements contrôlables peuvent être prohibés par le contrôleur alors que les événements incontrôlables peuvent se produire à tout moment.

Un système à événements discrets est contrôlé grâce à un vecteur de contrôle. Soit $\Gamma = \{0,1\}^{\Sigma_c}$, l'ensemble des applications $\gamma : \Sigma_c \to \{0,1\}$. Un vecteur de contrôle associe à chaque événement $\sigma \in \Sigma_c$ la valeur 1 si l'événement est permis ou la valeur 0 si l'événement est prohibé. On peut étendre γ à Σ en posant $\gamma(\sigma) = 1$ pour tout $\sigma \in \Sigma_u$, c'est-à-dire qu'un événement incontrôlable est toujours permis.

Un système à événements discrets contrôlé G_c est un automate fini déterministe $(Q, \Sigma\times\ \Gamma,\ \delta_c,\ q_0, Q_m)$ où : $\delta_c : Q\times\ \Sigma\ \times\ \Gamma \to Q$ est définie comme suit :

$$\delta_c(q,\sigma,\gamma) = \begin{cases} \delta(q,\sigma) & \text{si } \delta(q,\sigma) \text{ est définie et si } \gamma(\sigma) = 1 \\ \text{non définie} & \text{si } \delta(q,\sigma) \text{ est non définie ou si } \gamma(\sigma) = 0 \end{cases}$$

On remarque que l'automate G_c a le même comportement que G, sauf pour les événements contrôlables prohibés par le vecteur de contrôle.

Un contrôleur est un couple $C = (S, \phi)$, où $S = (X, \Sigma, \xi, x_0, X_m)$ est un automate fini déterministe et $\phi : X \to \Gamma$ est la fonction de rétroaction qui associe à chaque état du contrôleur un vecteur de contrôle et qui satisfait la propriété suivante :

$$\phi(x)(\sigma) = \begin{cases} 1 & \text{si } \sigma \in \Sigma_u, x \in X \\ 0 \text{ ou } 1 & \text{si } \sigma \in \Sigma_c, x \in X \end{cases}$$

Un système à événements discrets supervisé, noté $C \parallel G_c$, est défini comme suit : $Ac(X \times Q, \Sigma, \xi \times \delta_c, (x_0, q_0), X_m \times Q_m)$. La fonction de transition $\xi \times \delta_c : Q \times X \times \Sigma \to X \times Q$ est définie par

$$(\xi \times \delta_c)(q,x,\sigma) \ = \ \begin{cases} (\xi(x,\sigma), \delta(q,\sigma)) & \text{si } \xi(x,\sigma), \delta(q,\sigma) \text{ sont définies } \phi(x)(\sigma) = 1 \\ \text{non définie} & \text{autrement} \end{cases}$$

On exige souvent que le contrôleur C soit complet. Un contrôleur est complet si pour tout état (x, q) de $X \times Q$, $\delta(q, \sigma)$ est définie et $\phi(x)(\sigma) = 1$ implique que $\xi(x, \sigma)$ est définie.

Le problème de base de la théorie du contrôle des systèmes à événements discrets consiste à dériver, à partir du langage du procédé $L(G)$ et d'un langage légal $K \subseteq L(G)$, un contrôleur $C = (S, \phi)$ complet et le moins restrictif possible sur le comportement du procédé ($L(C \parallel G_c) \subseteq K$ est le plus grand langage possible). Le langage K est contrôlable par rapport à $L(G)$ et Σ_u si $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$. Cette condition spécifie que tout préfixe w de K concaténé avec un événement incontrôlable σ doit être également un préfixe de K ($w\sigma \in \overline{K}$) si la chaîne $w\sigma \in L(G)$.

Soit un langage légal $K \subseteq L(G)$, il existe un contrôleur $C = (S, \phi)$ tel que $L(C \parallel G_c) = K$ si et seulement si le langage légal K est fermé et contrôlable [35]. Il peut arriver que le langage légal ne soit pas contrôlable ; dans ce cas, on doit chercher le plus grand langage contrôlable, noté $K^{\uparrow}$, contenu dans le langage légal ($K^{\uparrow} \subseteq K$). D'après Ramadge et Wonham [35], ce langage existe et est unique.

Il existe un algorithme (voir figure 27), proposé par Wonham et Ramadge [48], qui permet de calculer $K^{\uparrow}$. À partir de $K^{\uparrow}$, il est possible de déduire le contrôleur $C = (S, \phi)$. L'automate S est l'automate qui accepte $K^{\uparrow}$. Dans cet algorithme, une fonction $h : X \rightarrow Q$ permet d'établir la correspondance entre les états de S et ceux de G. La fonction de rétroaction ϕ est définie comme suit. Soit $\Sigma(q)$, l'ensemble des événements possibles à partir de l'état q, c'est-à-dire $\Sigma(q) \subseteq \Sigma$ et $\sigma \in \Sigma(q)$ si $\delta(q, \sigma)$ est définie. Chaque événement $\sigma \in \Sigma(x)$ est permis ($\phi(x)(\sigma) = 1$). Soit $h(x) = q$ l'état correspondant à x dans G. Un événement contrôlable $\sigma \in \Sigma(q) - \Sigma(x)$ n'est pas permis ($\phi(x)(\sigma) = 0$). Le vecteur γ associé à l'état x est donné par $\gamma(\sigma) = \phi(x)(\sigma)$.

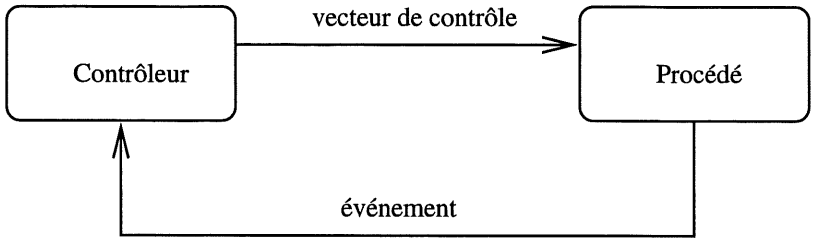

FIG. 26 – *Modèle simplifié d'une boucle de rétroaction*

1 **function** *Derive_Controller*(**in** G, H, Σ_u, **out** (S, ϕ))

/* G est l'automate du procédé et H l'automate du langage légal */

2 $\langle S, h\rangle \leftarrow G \times H$; /* $S = (X, \Sigma, \xi, x_0, X_m)$ */

3 **repeat**
4 $X' \leftarrow X$; $\xi' \leftarrow \xi$;
5 $X \leftarrow \{x \mid x \in X' \text{ and } \Sigma_u(h(x)) \subseteq \Sigma_u(x)\}$;
6 **for each** (x, σ) **in** $X \times \Sigma$ **do**
7 **if** $\xi'(x, \sigma) \in X$ **then**
8 $\xi(x, \sigma) \leftarrow \xi'(x, \sigma)$;
9 **else**
10 $\xi(x, \sigma)$ est indéfinie;

11 $Trim(S)$;
12 **until** $X = X'$;

13 **for each** x **in** X **do**
14 **for each** σ **in** $\Sigma - \Sigma_u$ **do**
15 **if not** $\xi(x, \sigma)!$ **then** $\phi(x) \leftarrow \phi(x) \cup \{\sigma\}$;
16 **end.**

FIG. 27 – *Algorithme de Wonham et Ramadge*

2.3.2 Approche par réseaux de Petri

Un réseau de Petri avec des places d'entrée externes ($PNIP$) est un modèle adéquat pour le contrôle de systèmes à événements discrets [45].

Le procédé (système à événements discrets) est modélisé à l'aide d'un réseau de Petri marqué $N = \langle R, M_0 \rangle$. Le langage légal est représenté par un prédicat Q sur l'ensemble des marquages accessibles de N, c'est-à-dire que Q indique quels sont, parmi tous les marquages accessibles de N, ceux qui sont franchissables.

On considère un réseau de Petri avec des places d'entrée externes $N_c = \langle R_c, M_{c_0} \rangle$, où $R_c = (P \cup P_c, T, I_c, O_c)$. Les ensembles $P = \{p_1, ..., p_n\}$ et $P_c = \{p_{c1}, ..., p_{cn}\}$ sont respectivement l'ensemble des places internes et l'ensemble des places externes ; T est l'ensemble des transitions ; $I_c : (P \cup P_c) \times T \to \mathbb{Z}$ (respectivement $O_c : (P \cup P_c) \times T \to \mathbb{Z}$) est le nombre d'arcs d'une place à une transition (respectivement d'une transition à une place) et $M_{c0} : (P \cup P_c) \to \mathbb{Z}$ est le marquage initial. Les contraintes suivantes sont imposées : $P \cap P_c = P \cap T = P_c \cap T = \emptyset$.

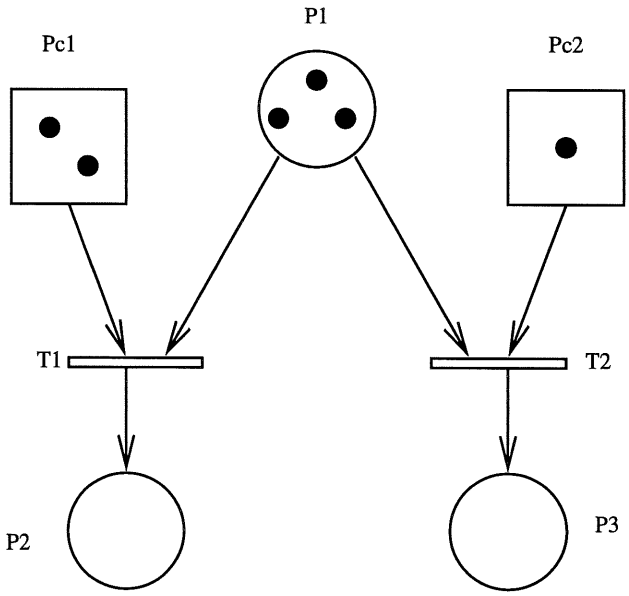

FIG. 28 – *Exemple simple d'un PNIP.*

La figure 28 montre un exemple d'un $PNIP$, où une place interne, une place d'entrée externe et une transition sont représentées respectivement par un cercle, un carré et une barre.

Puisque les places d'entrée externes supervisent le franchissement des transitions, on peut assumer sans perte de généralité que pour tout $p_c \in P_c$ et pour tout $t \in T$,

- $I_c(p_c, t) = 1$ ou 0 (c'est-à-dire, une transition consomme au plus un jeton),
- $O_c(p_c, t) = 0$ (aucun arc n'entre dans une place d'entrée externe).

Soit T_c et $T_u \subseteq T$ les ensembles de transitions dont les franchissements peuvent être respectivement contrôlés et non contrôlés par les places d'entrée externes :

$$T_c = \{t \in T : \exists p_c \in P_c \text{ tel que } I_c(p_c, t) = 1 \text{ et } p_c \in P_c\}$$

$$T_u = T \setminus T_c = \{t \in T \text{ et } t \notin T_c\}$$

Soit la fonction $\gamma : P_c \to \mathbb{Z}$ appelée vecteur de contrôle ; $\gamma(p_c)$ est le nombre de jetons associés à une place d'entrée externe p_c ($\forall p_c \in P_c M_c(p_c) = \gamma(p_c)$).

Soit $\Gamma = \mathbb{Z}^{P_c}$ l'ensemble des vecteurs de contrôle. Un système à événements discrets contrôlé G est un couple (N_c, Γ). Pour tout γ_1, $\gamma_2 \in \Gamma$, la somme $\gamma_1 \oplus \gamma_2$ et la multiplication $\gamma_1 \otimes \gamma_2$ sont définies comme suit. Pour tout $p_c \in P_c$,

$$(\gamma_1 \oplus \gamma_2)(p_c) = max\{\gamma_1(p_c), \gamma_2(p_c)\}$$

$$(\gamma_1 \otimes \gamma_2)(p_c) = min\{\gamma_1(p_c), \gamma_2(p_c)\}$$

Un « multi-set » (ou bag) sur T est appelé une *b–transition*. Soit T^w l'ensemble de toutes les *b–transitions*. Une *b–transition* $b \in T^w$ est dite franchissable pour un marquage M_c, notée par $M_c[b >$, si la condition suivante est vérifiée

$$\forall p \in P \cup P_c, \sum_{t \in b} I_c(p, t) \leq M_c(p)$$

Après le franchissement de b, M_c change en $M_c^{'}$, notée $M_c[b > M_c^{'}$, défini comme suit :

$$\forall p \in P,\ M_c^{'}(p) = M_c(p) + \sum_{t \in b}(O_c(p,t) - I_c(p,t))$$

À noter que le nombre de jetons associés à une place d'entrée externe est déterminé par la fonction de rétroaction (voir à la page 71).

Dans l'exemple de la figure 28, une *b–transition* $b = \{t_1, t_1, t_2\}$ est franchissable en raison de l'équation suivante :

$$\sum_{t \in b} I_c(p,t) = I_c(p,t_1) + I_c(p,t_1) + I_c(p,t_2) \leq M_c(p) \text{ pour } p = p_1, p_2, p_3, p_{c_1} \text{ et } p_{c_2}.$$

Après le franchissement de la transition $M_c^{'}(p_1) = 0, M_c^{'}(p_2) = 2, M_c^{'}(p_3) = 1$.

Soit T^{w^*} l'ensemble de toutes les suites de *b–transitions* de T^w incluant la séquence vide λ. Soit T_u^w et T_c^c les ensembles de toutes les « multi-sets », respectivement, sur T_u et T_c, et $S(\gamma)$ l'ensemble des *b–transitions* dont le franchissement peut être permis par le vecteur de contrôle γ :

$$S(\gamma) = \{b \in T^w : \sum_{t \in b} I_c(p_c,t) \leq \gamma(p_c) \text{ pour tout } p_c \in P_c\}$$

Il est évident que $T_u^w \subseteq S(\gamma)$ pour tout $\gamma \in \Gamma$.

Soit $N = \langle R, M_0 \rangle$, $R = (P, T, I, O)$, le réseau de Petri marqué réduit (sans les places externes) de N_c, c'est-à-dire que pour tout $p \in P$ et pour tout $t \in T$, les équations suivantes sont vérifiées :

$$I(p,t) = I_c(p,t)$$

$$O(p,t) = O_c(p,t)$$

$$M_0(p) = M_{c_0}(p)$$

Soit $T_b \subseteq T^w$ l'ensemble des *b–transitions* qui peuvent être franchissables pour un marquage accessible quelconque.

$$T_b = \{b \in T^w : M[b > \text{ pour } M \in A(R; M_0)\}$$

Une fonction de rétroaction $f \in \Gamma^{A(R;M_0)}$ est une application qui associe un vecteur de contrôle de Γ à chaque marquage accessible de N ($f : A(R;M_0) \rightarrow \Gamma$). Si M est le marquage de N alors le marquage M_c de la boucle de rétroaction (ou le système à événements discrets supervisé) $G \mid f$ est défini comme suit :

$$M_c(p) = M(p) \text{ pour chaque } p \in P$$

$$M_c(p_c) = f(M)(p_c) \text{ pour chaque } p_c \in P_c$$

Pour f_1 et $f_2 \in \Gamma^{A(R;M_0)}$, la somme $f_1 \oplus f_2$ et la multiplication $f_1 \otimes f_2$ sont définies par :

$$(f_1 \oplus f_2)(M) = f_1(M) \oplus f_2(M)$$

$$(f_1 \otimes f_2)(M) = f_1(M) \otimes f_2(M)$$

On peut aussi écrire $f(M) \in \Gamma$ comme un n-uplet :

$$((f(M)(p_{c_1})), (f(M)(p_{c_2})), \ldots, (f(M)(p_{c_n}))$$

Soit $\mathbf{Q} = \{0,1\}^{A(R;M_0)}$ l'ensemble des prédicats sur l'ensemble des marquages accessibles de N. Pour tout Q_1 et $Q_2 \in \mathbf{Q}$, la négation $\sim Q_1$, la conjonction $Q_1 \vee Q_2$ et la disjonction $Q_1 \wedge Q_2$ sur $\mathbf{Q}$ sont définies par :

$$\sim Q_1(M) = \begin{cases} 1 & \text{si } Q_1(M) = 0 \\ 0 & \text{si } Q_1(M) = 1 \end{cases}$$

$$(Q_1 \wedge Q_2)(M) = min\{Q_1(M), Q_2(M)\}$$

$$(Q_1 \vee Q_2)(M) = max\{Q_1(M), Q_2(M)\} \text{ pour tout } M \in A(R;M_0)$$

Une transformation $wlp_b : \mathbf{Q} \rightarrow \mathbf{Q}$ pour tout $b \in T^w$ est définie comme suit :

$$wlp_b(Q)(M) = \begin{cases} 0 & \text{si } M[b > M' \text{ et } Q(M') = 0 \\ 1 & \text{autrement} \end{cases}$$

La transformation wlp_b est appelée la précondition libérale la plus faible.

Contrôle et rétroaction

Pour chaque fonction de rétroaction $f \in \Gamma^{A(R;M_0)}$ et chaque *b–transition* $b \in T_b$, le prédicat f_b et la transformation wlp_b^f sur $\mathbf{Q}$ sont définis par :

$$f_b(M) = \begin{cases} 1 & \text{si } b \in S(f(M)), \\ 0 & \text{autrement} \end{cases}$$

$$wlp_b^f(Q) = wlp_b(Q) \vee \sim f_b$$

Un prédicat $Q \in \mathbf{Q}$ est dit contrôle–invariant (par rapport à G), s'il existe une fonction de rétroaction $f \in \Gamma^{A(R;M_0)}$ telle que, pour tout $b \in T_b$

$$Q \leq wlp_b^f(Q)$$

La fonction de rétroaction f est alors dite permissive pour Q, c'est-à-dire que dans un système à événements discrets supervisé $G \mid f$, Q est toujours vrai pour n'importe quel marquage accessible à partir de tout marquage satisfaisant Q.

Un ordre partiel « $\leq$ » sur $\Gamma^{A(R;M_0)}$ est défini comme suit : pour les fonctions de rétroaction f_1 et $f_2 \in \Gamma^{A(R;M_0)}$, $f_1 \leq f_2$ si seulement si $f_1(M)(p_c) \leq f_2(M)(p_c)$ pour tout $M \in A(R; M_0)$ et $p_c \in P_c$. Nous avons les résultats suivants [45] :

- Soit f_1 et $f_2 \in \Gamma^{A(R;M_0)}$, si $f_1 \leq f_2$, alors $S(f_1(M)) \subseteq S(f_2(M))$ pour chaque $M \in A(R; M_0)$, c'est-à-dire que $f_{1b} \leq f_{2b}$ pour tout $b \in T_b$.

- Soit f_1, f_2 et $f \in \Gamma^{A(R;M_0)}$

  - si $f_1 \leq f$ et $f_2 \leq f$, alors $f_1 \oplus f_2 \leq f$ et $f_1 \otimes f_2 \leq f$

  - si $f_1 \geq f$ et $f_2 \geq f$, alors $f_1 \oplus f_2 \geq f$ et $f_1 \otimes f_2 \geq f$

Soit la condition suivante, appelée *Minimum Token Condition* (MTC) : Pour tout $M \in A(R; M_0)$ et pour tout $p_c \in P_c$, $f(M)(p_c) = 0$ ou il existe un *b–transition* b tel que $M[b >$ et $f(M)(p_c) = \sum_{t \in b} I_c(p_c, t)$.

Une fonction de rétroaction permissive f satisfaisant MTC signifie que $f(M)$ affecte un nombre minimum des jetons pour chaque place d'entrée externe p_c franchissable.

En général, il existe plusieurs fonction de rétroaction permissive pour un prédicat de contrôle–invariant Q. Soit $\Omega(Q)$ l'ensemble des fonctions de rétroaction permissives satisfaisant MTC pour Q. L'ensemble $\Omega(Q)$ est un sous-ensemble de $\Gamma^{A(R;M_0)}$. Un élément maximal dans $\Omega(Q)$ est appelé la fonction de rétroaction permissive maximale pour Q. En général, il existe plus d'une telle fonction en raison de l'ordre partiel « $\leq$ ».

Toutefois, s'il existe seulement une transition $t \in T_c$ pour chaque place d'entrée externe $p_c \in P_c$ avec $I_c(p_c, t) = 1$ et s'il n'y a pas de concurrence (c'est-à-dire qu'une seule transition est franchie à la fois), alors il existe une fonction de rétroaction maximale permissive unique pour Q si Q est contrôle–invariant [45].

Chapitre 3

Exemple d'une bibliothèque

3.1 Description du problème

Une bibliothèque offre à ses abonnés des livres. Une personne qui désire emprunter des livres doit tout d'abord s'inscrire à la bibliothèque. Un abonné ne peut pas annuler son abonnement s'il possède des livres de la bibliothèque. Il doit d'abord rendre les livres qui sont en sa possession avant de quitter. Pour contrer le problème d'explosion combinatoire du nombre d'états, nous imposons la contrainte suivante : la bibliothèque ne gère que trois livres et deux abonnés.

3.2 Formalisation de la procédure d'affaires

Dans un premier temps, nous allons modéliser le comportement dynamique de chaque classe d'objets à l'aide d'un automate. Les classes de notre système sont les abonnés et les livres. Les événements propres aux abonnés (`utilisateur1 et utilisateur2`) sont : *inscrire*, *quitter*, *emprunter*, *renouveler* et *retourner*. Les événements propres aux livres (`livre1, livre2 et livre3`) sont : *acquérir*, *départir*, *emprunter*, *renouveler* et *retourner*. Seul l'événement *retourner* est incontrôlable.

Le comportement d'un abonné est spécifié à l'aide de l'automate de la figure 29, celui d'un livre à l'aide de l'automate de la figure 30.

Pour chaque abonné, il existe une instance de l'automate de la figure 29. Nous donnons ici l'automate de l'`utilisateur1` (voir figure 31). Les événements de cet automate se terminent tous par le chiffre 1, car ils correspondent à des actions de l'`utilisateur1`. Un événement se terminant par deux chiffres permet de distinguer sur quel livre (premier chiffre de la suite de deux chiffres) porte l'action de l'abonné. De façon similaire aux abonnés, il existe une instance de l'automate de la figure 30 pour chaque livre. La figure 32 donne l'automate pour le `livre1`. Le premier chiffre de la suite de chiffres à la fin des noms des événements réfère au `livre1` tandis que le deuxième chiffre réfère à l'`utilisateur1` ou l'`utilisateur2`.

Le comportement dynamique de la procédure d'affaires est obtenu par la composition synchrone de tous les automates (ceux des abonnés et ceux des livres).

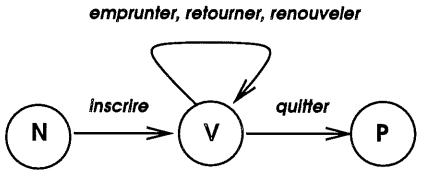

FIG. 29 – *Automate d'un abonné*

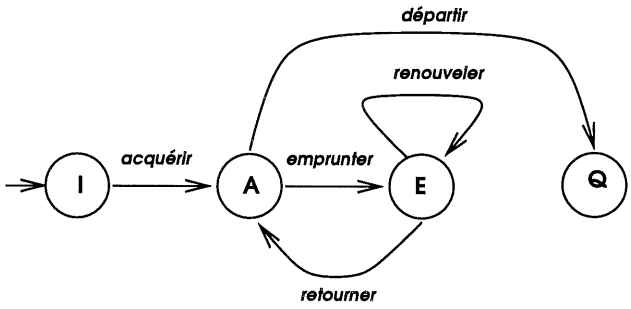

FIG. 30 – *Automate d'un livre*

```
AUTOMATON
utilisateur1
EVENTS: 11
ins1 qui1
emp11 ren11 ret11
emp21 ren21 ret21
emp31 ren31 ret31
END
CONTROLLABLE-EVENTS:
ins1 qui1
emp11 ren11
emp21 ren21
emp31 ren31
END
STATES: 3
N V P
END
MARKED-STATES:

END
INITIAL-STATE:
N
TRANSITIONS: 11
N : ins1  : V
V : qui1  : P
V : emp11 : V
V : ren11 : V
V : ret11 : V
V : emp21 : V
V : ren21 : V
V : ret21 : V
V : emp31 : V
V : ren31 : V
V : ret31 : V
END
```

FIG. 31 – *Automate de l'*`utilisateur1`

```
AUTOMATON
livre1
EVENTS: 8
acq1  dep1
emp11 ren11 ret11
emp12 ren12 ret12
END
CONTROLLABLE-EVENTS:
acq1  dep1
emp11 ren11
emp12 ren12
END
STATES: 5
I A E1 E2 Q
END
MARKED-STATES:

END
INITIAL-STATE:
I
TRANSITIONS: 8
I  : acq1  : A
A  : emp11 : E1
A  : emp12 : E2
A  : dep1  : Q
E1 : ren11 : E1
E1 : ret11 : A
E2 : ren12 : E2
E2 : ret12 : A
END
```

FIG. 32 – *Automate du* `livre1`

3.3 Règles de gestion

Deux règles de gestion sont imposées par les administrateurs de la bibliothèque :

- un abonné ne peut emprunter que deux livres au maximum ;
- un abonné doit rendre les livres en sa possession avant de quitter la bibliothèque.

La première règle de gestion est formalisée, pour l'`utilisateur1`, à l'aide de l'automate de la figure 33. Il faut noter qu'à chaque état, il y a des transitions vers le même état (*self-loop*) étiquetées par tous les événements qui n'apparaissent pas dans l'automate de la figure 33. Nous avons volontairement omis ces transitions afin de rendre le graphe lisible.

La deuxième règle de gestion est formalisée à partir du produit synchrone des automates des abonnés et des livres auquel on enlève les transitions étiquetées « *quitter* » à partir des états qui indiquent qu'au moins un utilisateur possible à au moins un livre.

Le langage légal est obtenu par l'intersection des langages acceptés par les automates de l'ensemble des règles de gestion.

3.4 Synthèse du contrôleur

Pour la synthèse du contrôleur de la procédure d'affaires à l'aide d'automates, nous utilisons le logiciel SUCSEDES développé à l'Université de Sherbrooke [32]. La figure 34 donne la fonction de rétroaction générée par l'outil SUCSEDES à partir de la composition synchrone des automates des abonnés et des livres et de l'intersection des automates des deux contraintes.

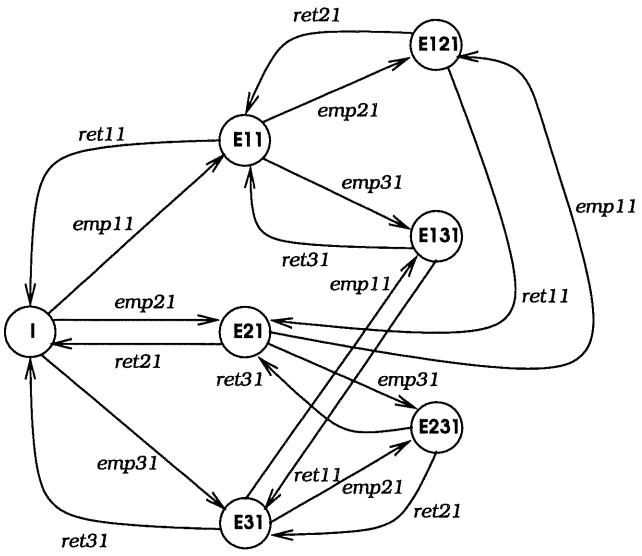

FIG. 33 – *Automate de la première contrainte*

```
( E1 E1 A V N E121 I E1 E1 A V N ) : { emp31 qui1 }
( E1 A Q V N E11 I E1 A Q V N ) : { qui1 }
( E1 A E1 V N E131 I E1 A E1 V N ) : { emp21 qui1 }
( E1 A A V V E11 I E1 A A V V ) : { qui1 qui2 }
( Q E1 A V N E21 I Q E1 A V N ) : { qui1 }
( A E1 Q V N E21 I A E1 Q V N ) : { qui1 }
( A E1 E1 V N E231 I A E1 E1 V N ) : { emp11 qui1 }
( A E1 A V V E21 I A E1 A V V ) : { qui1 qui2 }
```

FIG. 34 – *Fonction de rétroaction*

Par exemple, la première ligne signifie que l'`utilisateur1` ne peut pas quitter la bibliothèque car, il a au moins un livre en sa possession. Il ne peut pas emprunter le `livre3`, car il a déjà en sa possession deux livres (`livre1` et `livre2` représentés par `E121`), ce qui est le nombre maximum permis par la règle de gestion.

Chapitre 4

Exemple d'analyse des devis

4.1 Description du problème

La société de Sous-Traitance Mécanique (S.T.M.) est une P.M.E. qui réalise la fabrication et la vente de pièces mécaniques destinées principalement à l'automobile ou à l'aviation [36].

L'activité de l'entreprise porte sur la fabrication de pièces standards (vis, arbres) qui sont produites de façon continue et proposées par l'intermédiaire d'un catalogue qui fixe le tarif. Les pièces sont en général disponibles.

Cette société peut également fabriquer des pièces à la demande, auquel cas un devis devra être effectué tenant compte de la quantité de métal utilisé pour la réalisation ainsi que le cours du métal. En effet, la fabrication d'éléments en métaux précieux est possible sur demande.

Les acteurs qui composent l'organisation sont au nombre de cinq :

- la **direction** qui détermine les orientations de l'organisation ;
- le **secrétariat** général qui a un rôle de contrôle ;
- la **comptabilité** qui édite les factures et leurs réglements ;

- le service des **ventes** qui reçoit les demandes des clients ;
- la **production** qui procède à l'établissement des devis et à la fabrication des différentes pièces vendues.

Les matières premières sont proposées par des fournisseurs qui fixent les tarifs qui seront utilisés lors de l'établissement de devis. Une demande de tarif peut être adressée par la production au fournisseur lorsque les éléments nécessaires à la fabrication ne figurent pas sur les catalogues que le service a en sa possession.

La demande d'articles par un client est adressée au service des ventes. Deux cas peuvent alors se présenter.

- L'article est une pièce standard auquel cas une information sur le tarif de la pièce en question est envoyée au client qui doit alors confirmer sa commande. Les tarifs sont transmis régulièrement par la direction au service des ventes.
- L'article n'est pas une pièce disponible, une demande de devis est alors rédigée par le service des ventes et expédiée au service de la production qui procédera à un chiffrage sous forme de devis dans lequel, suivant le type de matériel demandé, peut intervenir le cours du métal nécessaire. Le cours du métal est transmis périodiquement par la direction à la production. Le devis, une fois établi, est expédié au service des ventes qui est chargé de l'adresser au client. Pour cela un complément d'information sur le client est nécessaire, il est fourni par la consultation d'un fichier.

Dans tous les cas, si le client décide de passer la commande, il devra la confirmer après réception soit du tarif, soit du devis. Sa confirmation de commande doit alors être adressée au service des ventes.

4.2 Formalisation du problème

1. **Schéma des flux :** Il permet une vision synthétique qui peut avoir été initialisée lors d'un schéma directeur sous la forme d'une carte de circulation des informations (voir figure 35).

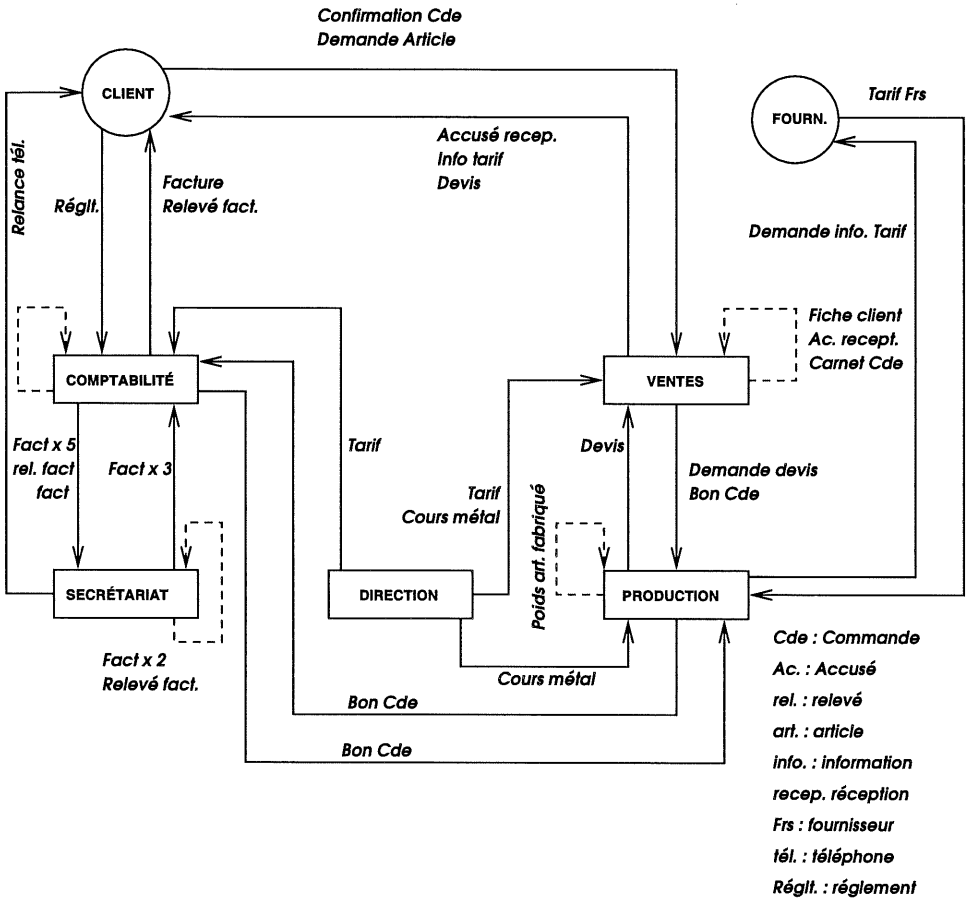

FIG. 35 – *Schéma des flux d'information*

2. **Graphe de dépendance des documents et identification des procédures d'affaires :** Ce graphe constitue une première mise en forme du schéma des flux. L'ensemble des documents qui y figurent est ordonnancé et un premier classement dans le temps fait apparaître la succession finalisée des documents (voir figure 37).

Cette succession de documents est constituée d'un premier document qui provient de l'extérieur et d'une suite de documents élaborée par des acteurs obéissant à une finalité de gestion ou de pilotage (contrôleur) identique qui se termine soit par :

- un ou plusieurs documents émis simultanément à destination d'un même acteur externe, avec passage du système d'information à un état de repos ;
- un passage du système d'information à un état de repos sans émission de document.

Chacune des suites ainsi établies constitue l'une des procédures d'affaires de l'organisation. Ces procédures d'affaires peuvent être vues comme une boîte noire sollicitée par un ou plusieurs événements externes et restituant à l'extérieur un ou plusieurs résultats.

3. **Diagramme de circulation des documents par procédure d'affaires :** L'établissement des diagrammes passe donc par l'identification des tâches puis par leur codification en expliquant globalement les actions exécutées pour chacune d'elles par utilisation des fonctions types complétées par des actions organisationnelles. Les quatre procédures d'affaires identifiées sont (voir figure 36) :
   - DEVIS,
   - COMMANDE-FACTURATION,
   - RÈGLEMENT,
   - STATISTIQUES DES DEVIS.

   Dans notre exemple, nous allons détaillé seulement la procédure d'affaires du DEVIS (voir figures 37 et 38).

   La figure 37 montre la circulation d'information relative à la procédure d'affaires DEVIS. Elle exprime la réalité de l'organisation telle qu'elle est vécue par les acteurs

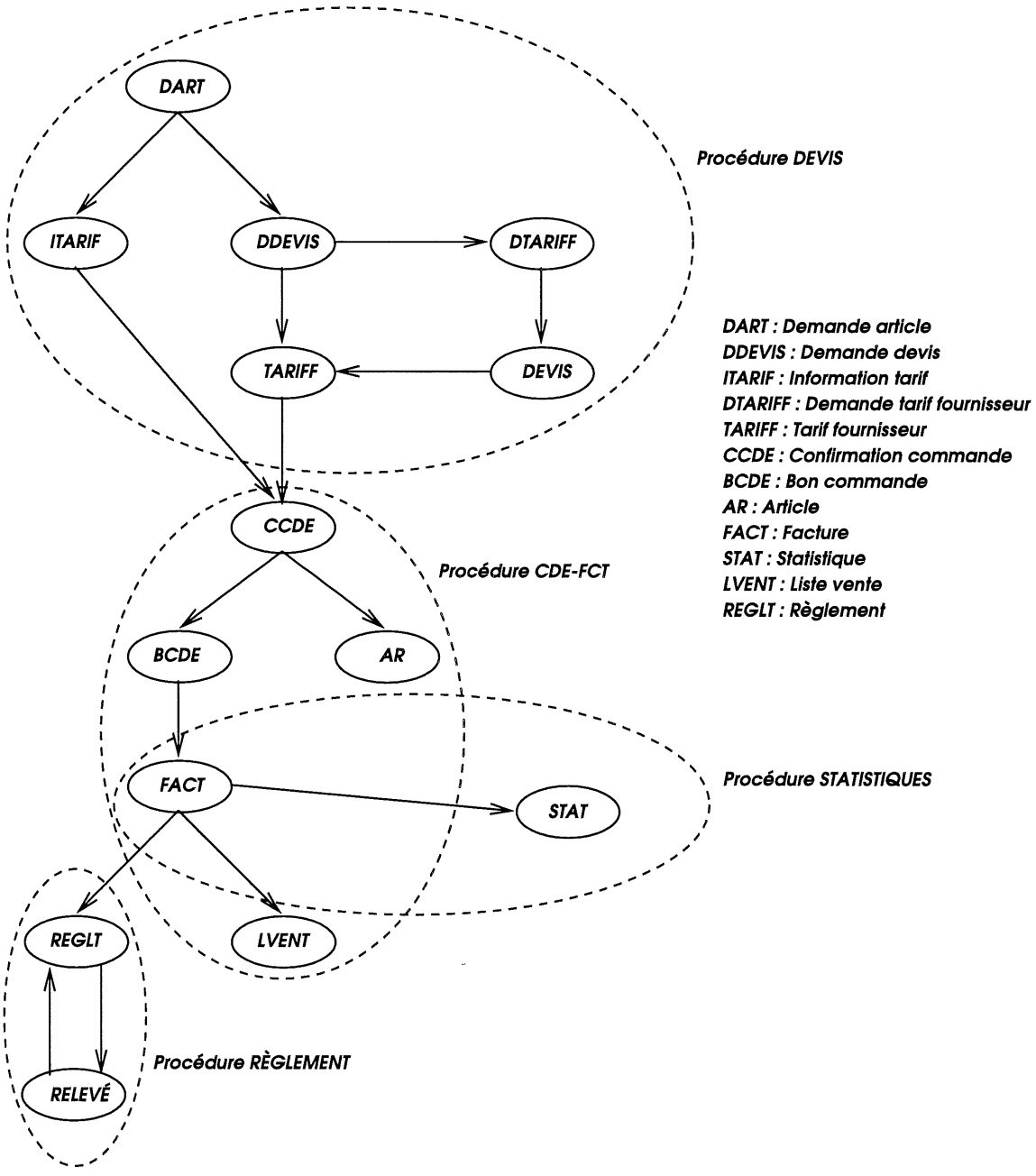

FIG. 36 – *Identification des procédures d'affaires*

de la procédure d'affaires. Ainsi, à ce niveau aucune différence n'est faite entre les hommes et les machines, pour autant que la dominance reste le *qui* et le *où* à l'exclusion du *comment.*

La figure 38 montre que les événements peuvent être séquentiels ou simultanés. Ce mode de cœxistance dans le temps est précisé par une synchronisation qui décrit une condition de franchissement.

4.3 Calcul du contrôleur

La procédure d'affaires DEVIS est modélisée à l'aide d'un réseau de Petri N (Voir figure 39). Le déclenchement de l'opération de consultation au sein du service de ventes nécessite l'arrivée de deux événements, une demande d'articles (DART) auprès du client et un catalogue des tarifs (TARIF) auprès de la direction. Deux résultats sont envisageables. Soit l'article est disponible (franchissement de la transition t_1) auquel cas une information sur le tarif est envoyée au client ; soit l'article est non disponible (franchissement de la transition t_2) auquel cas une demande de devis est envoyée au service de production qui déclenche une opération d'examen de demande de devis. Le résultat de l'examen donne (franchissement de la transition t_5) une demande de devis et une demande des tarifs qui sera envoyée au fournisseur. Une fois que le service de production a reçu les tarifs du fournisseur (franchissement de la transition t_6) et le cours du métal de la direction, une opération de création de devis est déclenchée en donnant comme résultat le devis qui sera envoyé ensuite au service de ventes. Ce dernier, est chargé de l'adresser au client (franchissement de la transition t_8).

Le système à événements discrets contrôlé est le couple (N_c, Γ), où N_c est un réseau de Petri avec une place d'entrée externe (voir figure 40) obtenu de N. Ainsi, $N_c = (P \cup P_c, T, I_c, O_c, M_{c0})$, où $P = \{p_1, \ldots, p_9\}$, $P_c = \{p_{c_1}\}$, $T = \{t_1, \ldots, t_9\}$.

L'ensemble des transitions est divisé en deux sous-ensembles T_c et T_u qui dénotent

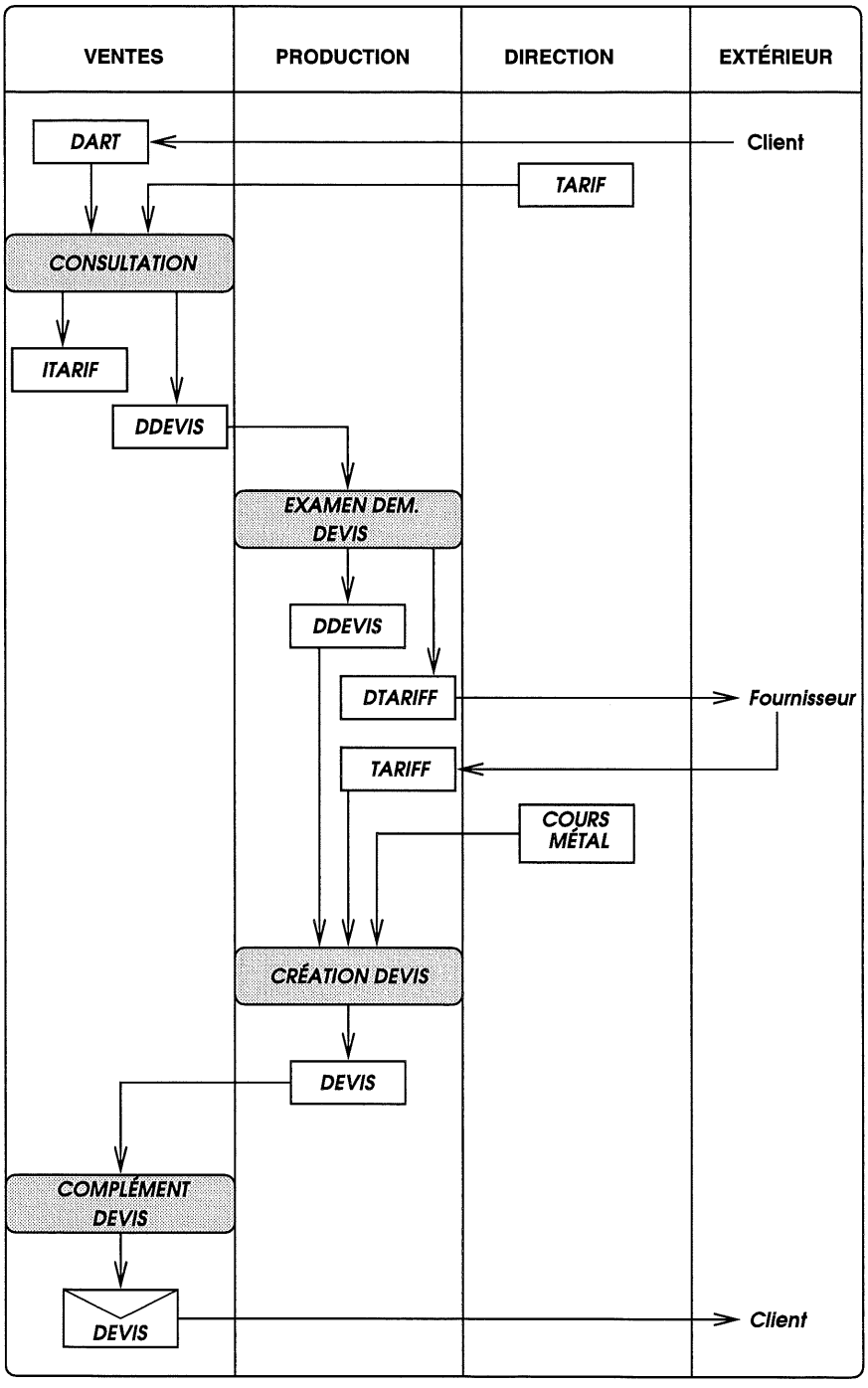

FIG. 37 – *Circulation des documents de la procédure d'affaires DEVIS*

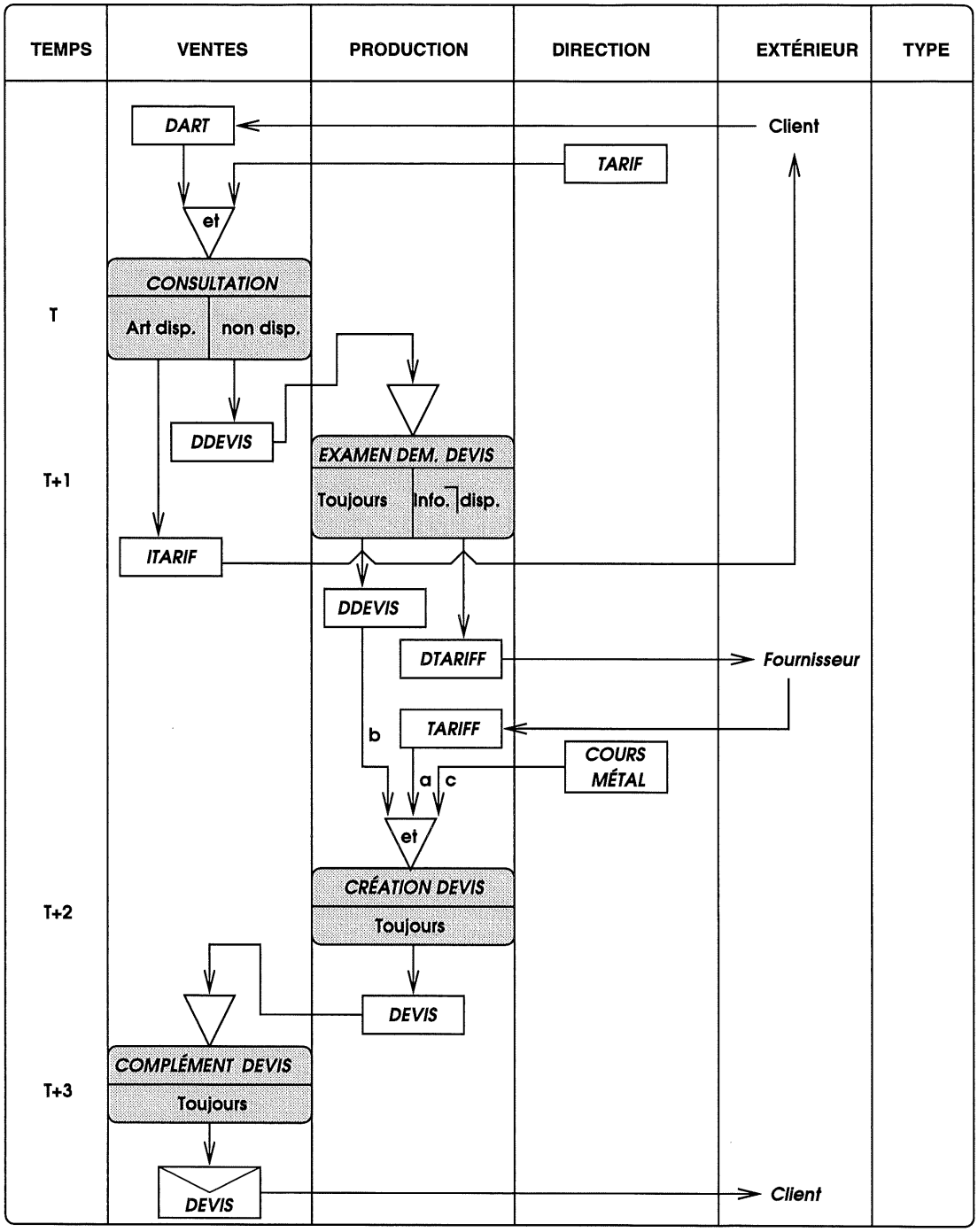

FIG. 38 – *Modèle de la procédure d'affaires DEVIS*

respectivement les ensembles des transitions dont les franchissements sont contrôlés et non contrôlés par les places d'entrée externes. Dans notre exemple,

$$\begin{aligned} T_c &= \{t_5\} \\ T_u &= \{t_1, t_2, t_3, t_4, t_6, t_7, t_8, t_9\} \end{aligned}$$

Dans notre exemple, le système de pilotage de l'organisation doit contrôler le nombre maximum de demandes de tarif au fournisseur pour un même devis. Il autorise, à l'aide d'une contrainte d'affaires (règle de gestion), au plus deux fois la demande de tarif au fournisseur. Cette règle de gestion est modélisée par un prédicat de contrôle Q défini comme suit :

$$Q(M) = \begin{cases} 1 & \text{si } M(p_5) + M(p_6) \leq 2 \\ 0 & \text{autrement} \end{cases}$$

Dans notre exemple, nous traitons juste une demande de devis à la fois vu que la représentation graphique à l'aide d'un réseau de Petri ordinaire ne permet pas de distinguer entre deux jetons représentant deux devis différents. Une extension à l'aide de réseaux de Petri colorés est toutefois possible [25].

Une condition pour montrer qu'il existe une fonction de rétroaction maximale permissive est de démontrer que le prédicat Q est contrôle–invariant (selon l'hypothèse faite à la fin de la section 2.3). Rappelons que le prédicat Q est contrôle–invariant, s'il existe une fonction de rétroaction $f \in \Gamma^{A(R;M_0)}$ telle que, pour tout $b \in T_b$

$$Q \leq wlp_b(Q) \vee \sim f_b$$

Prenons la fonction de rétroaction définie comme suit :

$$f(M)(p_{c_1}) = \begin{cases} 1 & \text{si } M(p_5) + M(p_6) \leq 1, \\ 0 & \text{autrement} \end{cases}$$

Nous injectons un jeton dans la place d'entrée externe si le nombre total de jetons dans les places p_5 et p_6 est égal 0 ou 1. Dans le cas contraire, nous laissons la place d'entrée externe vide.

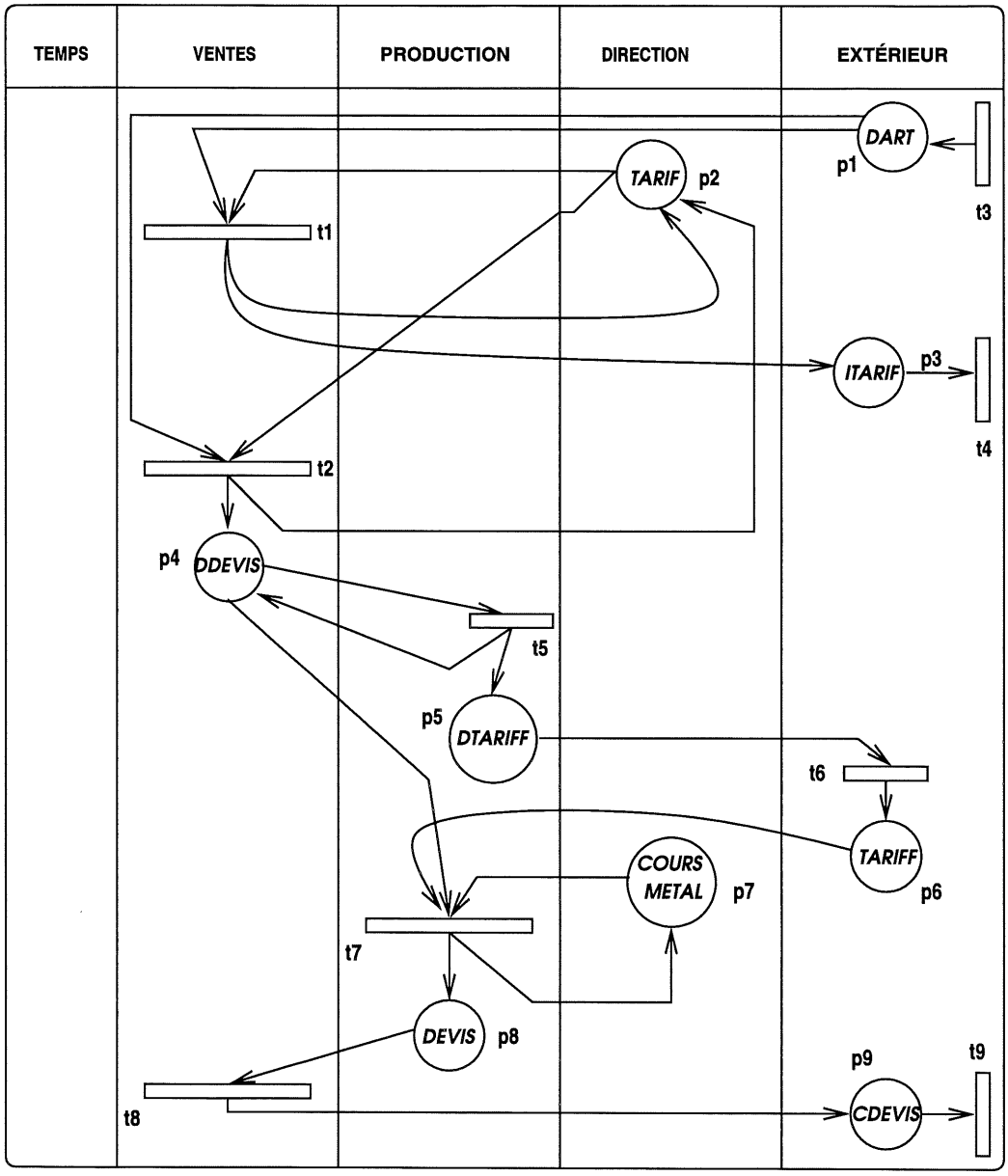

FIG. 39 – *Réseau de Petri de la procédure d'affaires DEVIS*

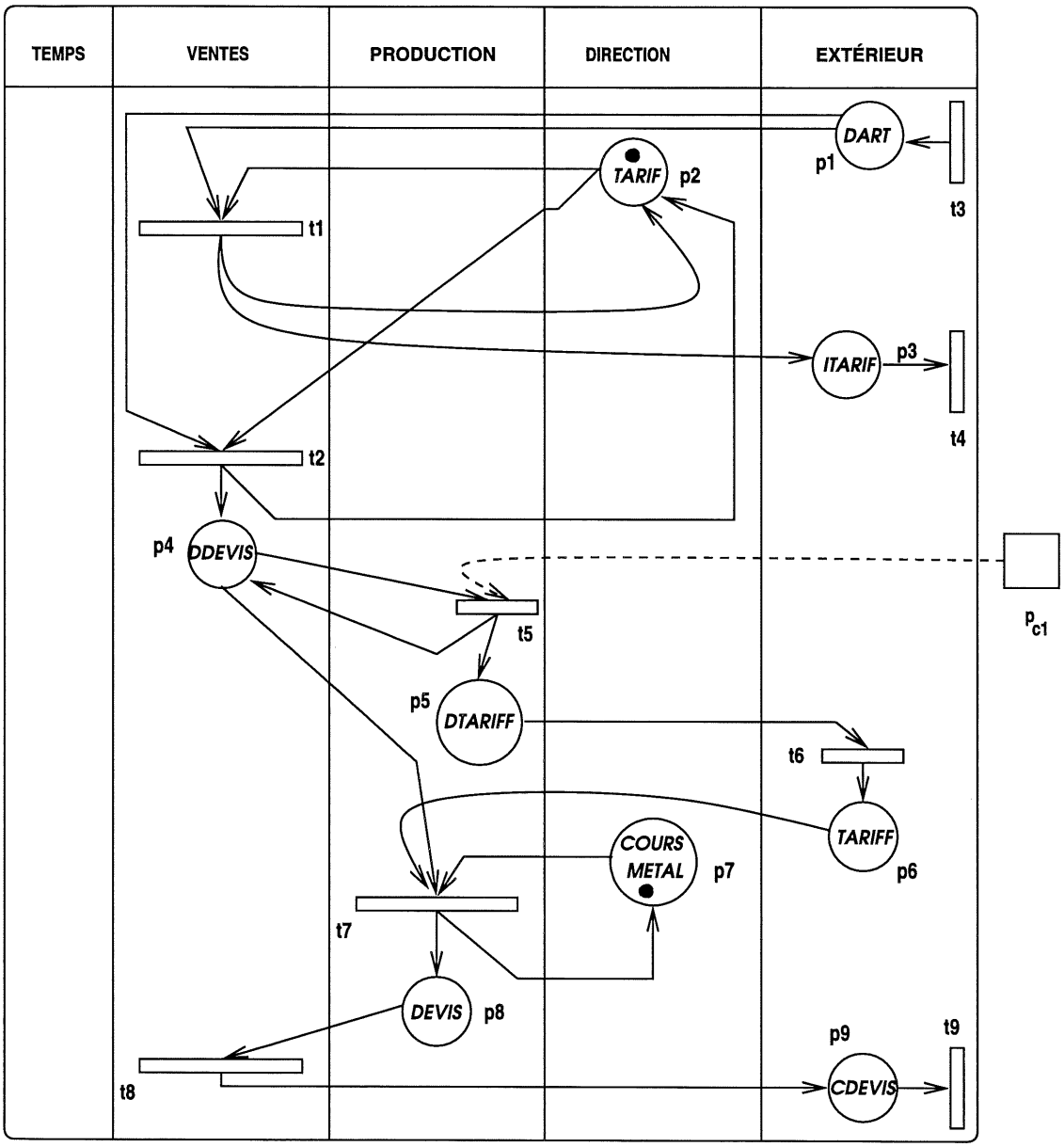

FIG. 40 – *PNIP de la procédure d'affaires DEVIS*

Intuitivement, f_b indique si la transition b est franchissable sous le contrôle de f. Comme t_5 est la seule transition que contrôle f, nous n'avons pas à nous préoccuper des autres transitions pour calculer f_b, car $f_b(M) = 1$ pour $b \neq t_7$. Lorsque $f_b(M) = 0$, alors $wlp_b^f(Q) = 1$ et donc $Q(M) \leq wlp_b^f(Q)(M)$. Si $Q(M) = 1$ et $f_b(M) = 1$, alors il faut que $wlp_b(Q)(M) = 1$ pour que Q soit contrôle–invariant quelque soit $M \in A(R; M_0)$. Ceci est facilement vérifiable en se référant au tableau 2 qui couvre toutes ces possibilités.

On pourrait aussi, toujours en se référant au tableau 2, montrer que f est la fonction de rétroaction la moins restrictive.

$\mathbf{M} \in \mathbf{A}(\mathbf{R};\mathbf{M_0})$	$\mathbf{f_b}(\mathbf{b} \neq \mathbf{t_5})$	$\mathbf{f_{t_5}}$	$wlp_b(Q)$									Q
			t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	
p_2p_7	1	1	1	1	1	1	1	1	1	1	1	1
$p_1p_2p_7$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_3p_7$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_4p_7$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_4p_5p_7$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_4p_5p_5p_7$	1	0	-	-	-	-	-	-	-	-	-	1
$p_2p_4p_5p_5p_5p_7\cdots$	1	0	-	-	-	-	-	-	-	-	-	0
$p_2p_4p_6p_7$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_4p_5p_6p_7$	1	0	-	-	-	-	-	-	-	-	-	1
$p_2p_4p_6p_6p_7$	1	0	-	-	-	-	-	-	-	-	-	1
$p_2p_4p_5p_5p_6p_7\cdots$	1	0	-	-	-	-	-	-	-	-	-	0
$p_2p_4p_5p_6p_6p_7\cdots$	1	0	-	-	-	-	-	-	-	-	-	0
$p_2p_4p_6p_6p_6p_7\cdots$	1	0	-	-	-	-	-	-	-	-	-	0
$p_2p_7p_8$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_5p_7p_8$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_6p_7p_8$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_5p_5p_7p_8\cdots$	1	0	-	-	-	-	-	-	-	-	-	1,0 $\cdots$
$p_2p_5p_6p_7p_8\cdots$	1	0	-	-	-	-	-	-	-	-	-	1,0 $\cdots$
$p_2p_6p_6p_7p_8\cdots$	1	0	-	-	-	-	-	-	-	-	-	1,0 $\cdots$
$p_2p_7p_9$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_5p_7p_9$	1	1	1	1	1	1	1	1	1	1	1	1
$p_2p_6p_7p_9$	1	1	1	1	1	1	1	1	1	1	1	1,0 $\cdots$
$p_2p_5p_5p_7p_9\cdots$	1	0	-	-	-	-	-	-	-	-	-	1,0 $\cdots$
$p_2p_5p_6p_7p_9\cdots$	1	0	-	-	-	-	-	-	-	-	-	1,0 $\cdots$
$p_2p_6p_6p_7p_9\cdots$	1	0	-	-	-	-	-	-	-	-	-	1,0 $\cdots$

TAB. 2 – *Les marquages accessibles*

Conclusion

Dans cette section, nous présentons d'abord un résumé des contributions de notre travail de recherche. Ensuite, nous critiquons l'approche utilisée dans ce travail. Enfin, nous dégageons une piste pour des travaux futurs de recherche.

Contributions

Les contributions de notre travail de recherche peuvent être regroupées en trois points :

- Nous avons montré comment les contrôleurs peuvent être intégrés à une organisation en substituant le système de pilotage au contrôleur et le système opérant au procédé.

- Nous avons choisi un outil adéquat de modélisation de comportements dynamiques des procédures d'affaires de l'organisation, qui est le modèle organisationnel des traitements (MOT) de la méthode Merise. En effet, le formalisme utilisé dans MOT a été initialement conçu en s'inspirant du formalisme des réseaux de Petri. C'est un formalisme intuitif et formel.

- Nous avons montré aussi à l'aide de deux exemples comment synthétiser des contrôleurs à l'aide d'algorithmes propres à la théorie du contrôle des systèmes à événements discrets. Le premier exemple porte sur une bibliothèque et utilise les automates comme outil formel de modélisation des comportements dynamiques d'une

procédure d'affaires et des règles de gestion. Le deuxième exemple porte sur le traitement de devis et utilise les réseaux de Petri comme outil formel de modélisation et le MOT de la méthode Merise.

Critiques du travail

Le choix des automates (machines à états finis) et des réseaux de Petri comme outils formels de modélisation du comportement des procédures d'affaires pose un certain nombre de problèmes.

Les automates ont un pouvoir d'expression limité puisqu'ils ne permettent que la représentation de langages réguliers. Ils ne prennent pas en compte l'explosion combinatoire qui les rend peu maniables pour les systèmes de contrôle. C'est pourquoi l'exemple de bibliothèque exploré dans ce mémoire est limité à la gestion de trois livres et de deux utilisateurs.

Les réseaux de Petri sont une formalisation de la concurrence et de la synchronisation des systèmes ayant une activité distribuée sans être liés à aucune notion de temps global. Bien qu'ils soient efficaces comme modèle conceptuel abstrait, ils se situent à un niveau trop bas et manquent d'expression pour être utiles dans la spécification des grandes organisations. C'est pour cela que nous avons choisi, pour modéliser le comportement dynamique des procédures d'affaires et des règles de gestion dans l'exemple d'analyse des devis, un outil de spécification informel comme MOT de Merise. Comme nous n'avons pas d'outil pour synthétiser des contrôleurs de procédures d'affaires modélisés à l'aide des réseaux de Petri, nous avons synthétisé manuellement le contrôleur.

Travaux futurs de recherche

Il existe actuellement un nouvel outil, appelé **Motown** [40], qui intègre l'analyse et la conception des procédures d'affaires (que doit-on faire? comment? qui est responsable?) ainsi que la génération automatique d'applications gérant les flux de travail (qu'est-ce qui est informatisé dans les procédures d'affaires?). Les procédures d'affaires sont conçues en utilisant les éditeurs graphiques spécialisés de **Motown**. Ces derniers permettent de produire automatiquement le système de gestion des flux de travail correspondant (Workflow Management System ou WFMS).

Une extension possible de notre travail de recherche est d'intégrer le contrôleur au système de gestion des flux de travail. En effet, l'outil **Motown** ne traite pas le contrôle des procédures d'affaires. Nous proposons une démarche possible présentée comme suit :

- modéliser le comportement dynamique des procédures d'affaires à l'aide d'OSSAD ;
- vérifier la mise en œuvre de la modélisation du comportement dynamique des procédures d'affaires à l'aide de l'outil **Motown** ;
- générer automatiquement le système de gestion des flux de travail à l'aide de l'outil **Motown** ;
- modéliser le comportement dynamique des règles de gestion de l'organisation ;
- générer un contrôleur à partir des procédures d'affaires et des règles de gestion ;
- développer une interface entre le système de gestion des flux de travail et le contrôleur généré à l'étape précédente.

Annexe A

Étude comparative des terminologies de Merise et de Workflow

La terminologie du workflow utilisée dans les tableaux (3, 4 et 5) est inspirée du glossaire de la Workflow Management Coalision (WfMC) [28]. Les tableaux contiennent les synonymes et leurs correspondants dans Merise [4] [12] [23] [26] [31] [44] des termes qui sont considérés comme les plus importants à ce stade du développement de la terminologie de la Workflow Management Coalision.

Terminologie workflow			Merise
Anglais	**Français**	**synonymes**	
Application Data	Données applicatives		
Business Process	Procédure d'entreprise	Business process model	procédure globale
Manual Process Activity	Activité manuelle	manual step, humain task, manual work	tâche manuelle
Manual Process Definition	Définition des procédures manuelles	physical routing definition, manual flow diagram, state transition diagram, flow schematic, manual script	diagramme des flux d'information
Manual Process Execution	Exécution d'une procédure manuelle		
Manual Process Instance	Cas de procédure manuelle		
Organisational Role	Rôle organisationnel	groupe d'utilisateurs	les usagers
Process	Procédure	Activity Network, directed graphe, Petri net, model	procédure fonctionnelle, opération organisationnelle
Process Activity	Activité	étape de travail ou de traitement ou de procédure, action, opération	opération
Process Activity Instance	Instance d'activité	step instance, node instance, task instance	tâche manuelle ou automatique
Process Definition	Définition des procédures	modèle des procédures, schéma des procédures, définition des routages, diagramme des flux, diagramme état/transition, carte des procédures	diagramme des flux d'information

TAB. 3 – *Étude comparative entre Workflow et Merise*

Terminologie workflow			Merise
Anglais	**Français**	**synonymes**	
Process Definition Mode	Modélisation des procédures	création et modification des procédures, conception des procédures, définition des procédures	procédure globale
Process Execution	Exécution d'une procédure	temps d'exécution	durée d'une procédure
Process Instance	Cas de procédure	cas d'exécution, instance de procédure, instanciation d'une procédure	procédure manuelle ou automatique
Process Role	Rôle procédural	groupe d'activités	groupe d'activités
Sub Process Définition	Définition des sous-procédures	subflow, subprocess	sous-procédure
Sub Process	Sous-procédure	Sub process definition, subflow	sous-procédure, procédure
Transaction Condition	Condition de transaction	conditions de branchement, règles de routage/de circulation	
Work Item	Bon de travail	tâche, unité de travail, élément de travail	tâche
Work Item Pool	Base des bons de travail	base des tâches, collection des tâches, réservoir des bons de travail, univers des bons de travail	les tâches
Workflow	Workflow	gestionnaire de flux de tâches, fluxgiciel, ordonnanceur	
Workflow Application	Application Workflow	application client, application, programme appelé	
Workflow Enactment Service	Service de Workflow	workflow automation, workflow manager, workflow computing system	

TAB. 4 – *Étude comparative entre Workflow et Merise (suite du tableau 3)*

Terminologie workflow			Merise
Anglais	**Français**	**synonymes**	
Workflow Interoperability	Intéropérabilité des Workflow		
Workflow Management System	Système de gestion de Workflow	système de gestion de flux de tâches, workflow	
Workflow Participant	Acteur du Workflow	agent, intervenant, participant, traitant, utilisateur	les ressources
Workflow Process	Procédure Workflow		
Workflow Process Activity	Activité Workflow	step, node, task, work element, process element, activity, workflow activity	tâche automatique
Workflow Process Activity Instance	Instance d'activité Workflow		
Workflow process Control Data	Données de contrôle des cas	données d'état du workflow, données de contrôle du workflow	
Workflow Process Définition	Définition des procédures Workflow	model definition, routing definition, flow diagram, state transition diagram, flow schematic, workflow script	diagramme des flux d'information
Workflow Process Engine	Moteur de Workflow	workflow management engine, workflow engine, coordination	
Workflow Process Execution	Exécution d'une procédure Workflow		
Workflow Process Instance	Cas de procédure Workflow		
Workflow Process Monitoring	Supervision des procédures		
Workflow Process Relevant Data	Données pertinentes pour les procédures	données d'état des procédures, données de contrôle des procédures	

TAB. 5 – *Étude comparative entre Workflow et Merise (suite du tableau 4)*

Bibliographie

[1] M. Ader, S. Murthy, N. Murthy, C. Bergnes, and J. Monguio. Organization model reference manuel. This paper was produced as part of the ESPRIT project 2705, ITHACA-2, 1992.

[2] M. Ader and P. Pons. Works's information model. This paper was produced as part of the ESPRIT project 2705, ITHACA-2, 1992.

[3] A. Agostini, G. De Michelis, and M. Grasso. The milano system. University of Milano, Italy, 1996.

[4] N. Alakl and J. Lalanne. *Architecture technologique des systèmes d'information: la méthode TACT.* Les éditions d'organisations, 1990.

[5] S. Benford. Requirements of activity management. In Studies in Computer Supported Cooperative Work, Theory Practice, and Design, J. Bowers and S. Benford Eds, North-Holland, 1991.

[6] C. Bignoli and C. Somine. I.A. techniques for supporting humain to humain communication in chaos. In Studies in Computer Supported Cooperative Work, Theory Practice, and Design, J. Bowers and S. Benford Eds, North-Holland, 1991.

[7] F. Bodard and T. Petitjean. Logistic cooperation : A model of decision making through other. Institut d'informatique, Facultés Notre-dame de la Paix, Namur, To appear in Journal of Decision Support System, 1996.

[8] G. Brams. *Réseaux de Petri : théorie et pratique tomes I et II.* Éditions Masson, 1983.

[9] CGI. Projet pac : programmation automatique corig. brochure technique, 1972.

[10] F. Cindio, G. Michelis, C. Simone, R. Vassalo, and A. Zaniboni. Chaos as a coordination technology. Proceedings of CSCW, 1986.

[11] W. David, P. Dumas, and al. Office support systems analysis and design : A manual. Esprit Projet 285, IOT. München, 1989.

[12] M. Daviné. *Parlez-vous merise.* EYROLLES, 1992.

[13] J. Dillmore and S. Wilbur. Experiences in building a configurable cscw systems. In Studies in Computer Supported Cooperative Work, Theory Practice, and Design, J. Bowers and S. Benford Eds, North-Holland, 1991.

[14] P. Dumas and G. Charbonnel. *La méthode OSSAD : pour maîtriser les technologies de l'information.* Les éditions d'organisation, 1990.

[15] C. A. Ellis and G. Nutt. Modelling and enactment of workflow systems, in : M. A. Marsan (Ed), Application and Theory of Petri Nets. Lecture Notes in Computer Science 691, Springer-Velag, 1993.

[16] Third european workshop of software process technology. EWSPT J. C. Warboys (Ed). Lecture Notes in Computer Science 772, Springer-Velag, 1994.

[17] F. Fritz. Workflow implementation based on the r/3 reference model. Report/manuscript, SAP AG, 1995.

[18] H. Heckenroth, H. Tardieu, and B. Espinasse. Modèles et outils pour la conception de la cinématique d'un système d'information. Rapport de recherche INRIA 310, CTE d'Aix-GRASCE/CNRS Universite d'aix-Marseille III, 1980.

[19] D. Hollingsworth. The workflow reference model. Technical Report TC00-1003, Workflow Management Coalition, 1994.

[20] D. Karagiannis. Towards business process management systems: A tutorial. Presented at the Second International Conference on Cooperative Informations Systems CoopIS, 1994.

[21] L. Lamport and N. Lynch. Distributed computing: models and methods. *MIT Press/Elsevier Cambridge*, pages 1157–1199, 1990.

[22] J. Longchamp. Supporting social interaction activities of software process. Lecture Notes in Computer Science 635, Software Technology, J. C. Derniame (Ed), Springer-Velag, 1992.

[23] G. Louvet and F. Meutre. *Se former à Merise: la modélisation conceptuelle.* Les éditions d'organisations, 1990.

[24] W. M. and V. Der Aalst. A class of Petri net for modeling and analyzing business processes. Computing Science Reports 26, Eindhoven University of Technology, 1995.

[25] M. Makungu, R. St-Denis, and M. Barbeau. A colored petri net-based approach to the design of controllers. In *Proc. of the 35th IEEE Conference on Decision and Control*, pages 4425–4432. Kobe, 1996.

[26] J. Matheron. *Comprendre Merise outils conceptuels et organisationnels.* EYROLLES, 1994.

[27] R. Medina-Mora, T. Winograd, R. Flores, and F. Flores. The action workflow approch to workflow management technology. Proceedings of CSCW, 1992.

[28] Les membres de WfMC. Une spécification de la workflow management coalition. Glossaire, 1996.

[29] G. Michelis. Organizational structures and cooperation support: Managing complexity. University of Milano, 1995.

[30] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[31] D. Nanci, B. Espinasse, B. Cohen, and H. Hecknoroth. *Ingénierie des systèmes d'information avec Merise.* SYBEX, 1992.

[32] J. M. Palmier, M. Barbeau M. Makungu, F.E. Agapi, and R. St-Denis. An introduction to synchronized Petri net based tool for the synthesis of supervisors of discrete-event systems. In *Proceedings of BMW*, pages 67–78. Méthode mathématiques pour la synthèse des systèmes informatique, 15, UQAM, 1994.

[33] T. PetitJean. *Contribution à la specification de situation de cooperation ad hoc et à leur prise en compte dans les systèmes de workflow.* PhD thesis, Facultés Notre-Dame de la Paix, Namur, Belgium, 1994.

[34] W. Poos. *Prise en compte de situations de coopération prédéfinies dans un système de workflow.* PhD thesis, Facultés Notre-Dame de la Paix, Namur, Belgium, 1994.

[35] P. J. Ramadge and W. H. Wonham. Supervisory control of a class of discret event processes. *Journal Control and Optimization*, 25(1):206–230, 1987.

[36] A. Rochfeld and J. Moréjon. *La méthode Merise.* Les éditions d'organisation, 1993.

[37] T. Schael. Supporting cooperative process with workflow mangement technology. Tutorial notes COOCS, 1993.

[38] J. Searle. Speech acts. Cambridge University Press, 1969.

[39] Site. http://www.daimi.aau.dk/petrinets/research/. Internet.

[40] Site. http://www.unil.ch/idheap/. Internet.

[41] H. Smith, P. Hennesy, and G. Lunt. An objet oriented framework for modelling organisational communication. In Studies in Computer Supported Cooperative Work, Theory Practice, and Design, J. Bowers and S. Benford Eds, North-Holland, 1991.

[42] S. Soubbaramayer. Les techniques du workflow. MOS 127, magazine, 1994.

[43] Numéro spécial Réseaux de Petri. Technique et sciences informatique. *Dunod-AFCET informatique*, 4(1), 1985.

[44] H. Tardieu, A. Rochfeld, and R. Colletti. *La méthode Merise Tome1 principes et outils.* Les éditions d'organisations, 1983.

[45] T. Ushio. Maximally permissive feedback and modular control synthesis in Petri nets with external input places. *IEEE Transactions on Automatic Control*, 35(7):844–848, 1990.

[46] K. Venkataramen. Cool source file: timeobj.t. This software was produced as part of the ESPRIT project 2705, ITHACA-2, 1992.

[47] T. While and L. Fischer. *New Tools for New Times: The Workflow Paradigm–The Impact of Information Technology on Business Process Reegineering.* Futur Strategies Inc. Book Division. Alameda. CA, 1994.

[48] W. H. Wonham and P. J. Ramadge. On the supremal controllable sublanguage of a given language. *Journal Control and Optimization*, 25(3):637–659, 1987.

[49] M. C. Zhou and F. DiCesare. *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems.* Kluwer Academic Publishers, 1993.